Teradactyl®

TiBS 2.0

True incremental Backup System®

VERSION 2.0

with TeraMerge® technology

User Manual

© Copyright 1997 – 2008, Teradactyl LLC. All Rights Reserved. TeraMerge® under patent.

True incremental Backup System® (TiBS)

And

True incremental Backup System® (TiBS) Lite 2.0

And

True incremental Backup System® (TiBS) Small Business Edition 2.0

User Manual

Teradactyl LLC.

Manual Version 2.1.0.9c
Copyright © 1997 - 2008 Teradactyl LLC.
All Rights Reserved. TeraMerge® now under U.S. Patent.
Printed in U.S.A.

The entire risk of the use or the result of the use of this software and documentation remains with the user. No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, except as expressed in the Software License Agreement.

Copyright © 1997 - 2008, Teradactyl LLC. All rights reserved. This software and documentation are copyrighted. All other rights, including ownership of the software, are reserved to Teradactyl®. This software is protected by one or more patents or pending patents. Teradactyl®, True incremental Backup System®, and TeraMerge® are all registered trademarks of Teradactyl LLC. Microsoft® and Windows® are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. AFS® is a registered trademark of IBM Corporation. UNIX® is a registered trademark licensed exclusively through X/Open Company Limited. Mac is a trademark of Apple Computer, Inc., registered in the U.S. and other countries. Reprise License Manager™ (RLM) is a trademark of Reprise Software, Inc. Open Transaction Manager™ is a trademark of Columbia Data Products, Inc. All other brand and product names are trademarks or registered trademarks of their respective owners.

Teradactyl LLC.
2301 Yale Blvd. S.E., Suite C-7
Albuquerque, New Mexico 87106-4352
Phone: (505) 242-1091

The True incremental Backup System® (TiBS), TiBS Lite, and TiBS Small Business Edition End-User Software License Agreement

IMPORTANT – READ CAREFULLY: This Software License Agreement ("SLA") is a legal agreement between you, the end user (either an individual or single entity), and Teradactyl LLC. Teradactyl® is the owner of a certain proprietary software package known as True incremental Backup System® (TiBS), which includes computer software and may include associated media, printed materials, and "online" or electronic documentation ("Product"). Use of the software indicates your acceptance of these terms. An amendment or addendum to the SLA may also accompany the Product.

YOU AGREE TO BE BOUND BY THE TERMS OF THE SLA BY INSTALLING, COPYING, OR OTHERWISE USING THE PRODUCT. IF YOU DO NOT AGREE TO THESE TERMS AND CONDITIONS, DO NOT INSTALL OR USE THE PRODUCT; EITHER DESTROY OR RETURN, INTACT, THE TRUE INCREMENTAL BACKUP SYSTEM® (TiBS) PACKAGE CONTAINING THE CD OR OTHER MEDIA, TOGETHER WITH THE OTHER COMPONENTS OF THE PRODUCT TO TERADACTYL® FOR A FULL REFUND WITHIN 30 DAYS OF PURCHASE. As used in this Software License Agreement, the term "SOFTWARE" means the object code version of the True incremental Backup System® (TiBS) software shipped either electronically, on CD, magnetic or disk media provided with this License Agreement. The term "SOFTWARE" does not include any software that is covered by a separate license offered or granted by a person other than Teradactyl®. The Product may contain the following SOFTWARE:

"Server Software" provides services or functionality on your backup server.
"Client Software" allows an electronic device ("Device") to access or utilize the Server Software.

PROPRIETARY RIGHTS. The SOFTWARE and any accompanying documentation are proprietary products of Teradactyl or its licensors and are or may be protected under United States copyright laws and international treaty provisions. Ownership of the SOFTWARE and all copies, modifications, and merged portions thereof shall at all times remain with Teradactyl or its' licensors.

PATENTS AND INVENTIONS. Customer acknowledges that in the course of its performance hereunder, Teradactyl may use products, materials and methodologies proprietary to Teradactyl or to any other third party with which Teradactyl has authorization to use such products, materials, or methodologies. Customer agrees that it shall have or obtain no rights in such proprietary products, materials and methodologies, unless the parties enter into a separate written agreement to that effect.

GRANT OF LICENSE. The SOFTWARE and accompanying documentation are being licensed to you, which means you have the right to use the SOFTWARE only in accordance with this License Agreement. Teradactyl grants you the right to use the number of copies of the SOFTWARE specified in this license on a specific computer or computers. The SOFTWARE is considered in use on a computer when it is loaded into temporary memory or installed into permanent memory.

OPERATING SYSTEM-SPECIFIC LICENSE. The license is granted for each TiBS backup server operating system purchased and can be used on an unlimited number of server and client machines for each operating system licensed. The SOFTWARE may be used only on operating systems supported by Teradactyl and licensed to you. The SOFTWARE may only be used on computers (either stand-alone computers or computers connected to a network) owned or leased by you. Once a copy of the SOFTWARE has been used on any computer, that computer may not be sold, transferred, leased or loaned any other entity, department, or person, unless you have permanently stopped using (e.g., destroyed or relinquished possession of) the SOFTWARE and have removed the SOFTWARE from the original computer.

EDITION-SPECIFIC LICENSE. The license is granted for a specific edition of the True incremental Backup System. Each TiBS backup server license and processing pack purchased is required to be deployed with the TiBS edition sold to the end-user. The SOFTWARE, including server operating system licenses and processing packs, including version upgrades, is not transferable to other editions of TiBS.

SERVER PROCESSING PACK(S) LICENSE. Teradactyl grants you the right to use the number of processes sold for the SOFTWARE and edition specified in this license on a specific computer or computers. The SOFTWARE is considered in use on a computer when it is loaded into temporary memory or installed into permanent memory. Processes may be distributed among multiple TiBS servers of identical editions, but the total number of processes deployed may never exceed the number purchased by the end-user.

PERSONAL LICENSE. This license is personal to you. You may not sublicense, lease, sell, or otherwise transfer the SOFTWARE or any of the accompanying documentation to any other person, organization or entity. You may use the SOFTWARE only for your own personal use if you are an individual, or for your own internal business purposes if you are a business.

NONPERMITTED USES; CONFIDENTIALITY. Without the express permission of Teradactyl, you may not (a) use, copy, modify, alter, or transfer, electronically or otherwise, the SOFTWARE or documentation except as expressly permitted in this License Agreement, or (b) translate, reverse program, disassemble, decompile, or otherwise reverse engineer the SOFTWARE. Under no circumstances are you permitted to make the software or modified version therefrom, available to any third parties, including, without limitation, other departments within the organization, business, or government agency or on a network file server, without Teradactyl's prior written consent. You agree to take all necessary precautions to protect the confidentiality of the SOFTWARE. Under no circumstances are you permitted to assign or transfer any of its rights under this agreement.

TERM. This license is effective from your date of purchase and shall remain in force until terminated. You may terminate the license and the License Agreement at any time by destroying the SOFTWARE and the accompanying documentation, together with all copies in any form.

EXPORT CONTROLS. Certain uses of the SOFTWARE by you may be subject to restrictions under U.S. regulations relating to exports and ultimate end uses of computer software. You agree to fully comply with all applicable U.S. laws and regulations, including but not limited to the Export Administration Act of 1979 as amended from time to time and any regulations promulgated thereunder.

U.S. GOVERNMENT RESTRICTED RIGHTS. If you are acquiring the SOFTWARE on behalf of any unit or agency of the United States Government, the following provision applies: It is acknowledged that the SOFTWARE and the documentation were developed at private expense and that no part is in the public domain and that the SOFTWARE and the documentation are provided with RESTRICTED RIGHTS. Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Right in Technical Data and Computer Software clause at DFARS 252.227-7013 or subparagraphs (c)(1) and (2) of the Commercial Computer Software-Restricted Rights at 48 CFR 52.227-19, as applicable. Contractor/manufacturer is Teradactyl LLC. 2301 Yale Blvd. S.E., Suite C-7, Albuquerque, New Mexico, 87106.

LIMITED WARRANTY. (a) Teradactyl warrants to you, the original end user, (I) that the SOFTWARE, other than third-party software, will perform substantially in accordance with the accompanying documentation and (II) that the SOFTWARE is properly recorded on the disk media. This Limited Warranty extends for ninety (90) days from the date of purchase. Teradactyl does not warrant any third-party software that is included in the SOFTWARE, but Teradactyl agrees to pass on to you any warranties of the owner or licensor to the extent permitted by the owner or licensor. (b) This Limited Warranty does not apply to any SOFTWARE that has been altered, damaged, abused, misapplied, or used other than in accordance with this License and any instructions included on the SOFTWARE and the accompanying documentation. (c) Teradactyl's entire liability and your exclusive remedy under this Limited Warranty shall be the repair or replacement of any SOFTWARE that fails to conform to this Limited Warranty or, at Teradactyl's option, return of the price paid for the SOFTWARE. Teradactyl shall have no liability under this Limited Warranty unless the SOFTWARE is returned to Teradactyl or its authorized representative, with a copy of your receipt, within the warranty period. Any replacement SOFTWARE will be warranted for the remainder of the original warranty period or 30 days, whichever is longer.

LIMITATION OF LIABILITY. TERADACTYL DOES NOT UNDER THIS AGREEMENT WARRANT THE UNINTERRUPTED PERFORMANCE OF THE SOFTWARE PACKAGE OR THAT THE FUNCTIONS CONTAINED IN OR PERFORMED BY THE SOFTWARE PACKAGE WILL MEET YOUR SPECIFIC REQUIREMENTS. THE SOFTWARE PACKAGE IS PROVIDED TO YOU HEREUNDER "AS-IS", AND TERADACTYL PROVIDES NO WARRANTIES OR REPRESENTATIONS IN RESPECT OF THE SOFTWARE PACKAGE OR ANY DOCUMENTATION RELATING THERETO, EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF

MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WHETHER PROVIDED AT LAW OR IN EQUITY. TERADACTYL SHALL NOT IN ANY CASE BE LIABLE FOR SPECIAL, INCIDENTAL, CONSEQUENTIAL, INDIRECT OR OTHER SIMILAR DAMAGES ARISING FROM BREACH OF CONTRACT, NEGLIGENCE, OR ANY OTHER THEORY, EVEN IF TERADACTYL OR AN AUTHORIZED AGENT OF TERADACTYL HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES, WHETHER SUCH LIABILITY IS BASED ON CONTRACT, TORT, WARRANTY, OR ANY OTHER LEGAL OR EQUITABLE GROUNDS. BECAUSE SOME STATES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF LIABILITY FOR CONSEQUENTIAL OR INCIDENTAL DAMAGES, THE ABOVE LIMITATION MAY NOT APPLY TO YOU. THIS MEANS THAT TERADACTYL SHALL NOT BE RESPONSIBLE FOR ANY COSTS INCURRED AS A RESULT OF LOST PROFITS OR REVENUE, LOSS OF USE OF THE SOFTWARE PACKAGE OR ANY PORTION THEREOF, LOSS OR CORRUPTION OF DATA, COSTS OF RECREATING LOST OR CORRUPTED DATA, THE COST OF ANY SUBSTITUTE SOFTWARE, CLAIMS BY ANY PARTY OTHER THAN YOU, OR FOR OTHER SIMILAR COSTS. IN NO EVENT SHALL TERADACTYL'S LIABILITY RELATED TO ANY OF THE SOFTWARE EXCEED THE LICENSE FEES ACTUALLY PAID BY YOU FOR THE SOFTWARE EXCEPT FOR A RETURN OF THE PURCHASE PRICE UNDER THE CIRCUMSTANCES PROVIDED UNDER THE LIMITED WARRANTY.

INJUNCTIVE RELIEF. In the event of any actual or alleged breach by you of this agreement, Teradactyl, in addition to and not in limitation of any other remedies available to him hereunder or at law, shall be entitled to temporary or injunctive or other equitable relief.

GENERAL MATTERS:

CAPTIONS. The captions utilized in this Agreement are for the purposes of identification only and shall not control or affect the meaning or construction of any provisions hereof.

BENEFITS AND BINDING EFFECT. This Agreement shall be binding upon and shall inure to the benefit of each of the parties and of their respective successors and permitted assigns.

INTEGRATION. This Agreement constitutes the entire agreement between the parties with respect to the subject matter hereof and supersedes all previous negotiations, representations, commitments and writings.

MODIFICATION AND WAIVER. This Agreement may not be amended, released, discharged, rescinded or abandoned, except by a written agreement duly executed by each of the parties hereto. The failure of any party hereto at any time to enforce any of the provisions of this Agreement will in no way constitute or be construed as a waiver of such provision or of any other provision hereof nor, in any way affect the validity of, or the right thereafter to enforce, each and every provision of this Agreement.

GOVERNING LAW. This Agreement and its validity, construction, administration, and all rights hereunder, shall be governed by the laws of the United States of America, State of New Mexico without regard to its conflict of laws provisions. Any litigation arising from this license will be pursued only in the state or federal courts located in the United States of America, State of New Mexico.

SEVERABILITY. The invalidity or unenforceability of any particular provision of this Agreement shall not affect the other provisions hereof, and this Agreement shall be construed in all respects as if such invalid or unenforceable provisions were omitted.

COUNTERPARTS. This Agreement may be executed in several counterparts, each of which shall be deemed to be an original, but all of which shall constitute one and the same instrument.

For further information: Should you have any questions concerning this Agreement or if you desire to contact Teradactyl for any reason, please write:

Teradactyl LLC.
2301 Yale Blvd. S.E., Suite C-7
Albuquerque, New Mexico 87106-4352
E-Mail: info@teradactyl.com

Table of Contents

TABLE OF CONTENTS **7**

WELCOME TO TIBS 2.0 ..... **12**

INTRODUCTION **13**

HOW TRADITIONAL BACKUP SYSTEMS WORK 13

HOW TERAMERGE® WORKS 13

New Disk Library Interface (DLI) *14*

Types of Backup Available with TiBS: *14*

REDUNDANCY CONSIDERATIONS IN TIBS 15

CONVENTIONS USED IN THE DOCUMENT 16

MANUAL OVERVIEW 17

INSTALLATION **18**

PRE-INSTALL REQUIREMENTS 18

INSTALL OPTIONS 18

SERVER INSTALLATION 19

RLM License File *19*

Uninstalling the TiBS Server *20*

CLIENT INSTALLATION FOR UNIX® 20

Client Install Options *20*

Installing UNIX® Clients from the Backup Server *21*

Installing a Single Client 21

Installing Multiple Clients 21

Uninstalling UNIX® Clients *22*

INSTALLING WINDOWS® CLIENTS 22

Upgrading/Uninstalling Windows® Clients *24*

Installing TiBS via Group Policy *25*

Copy the TiBS installation media to a network share 25

Customize tibs.msi with site defaults 25

Create TiBS installation policy 26

Installing Windows® Clients Remotely *27*

SERVER CONFIGURATION **28**

STARTING AND STOPPING THE SERVER 28

CONFIGURATION FILES OVERVIEW 29

CONFIGURATION OF TIBS.CONF 30

Settings for tibs.conf use with AFS® *36*

UNIX® Program Pathnames Defined in tibs.conf *38*

DEFINING THE BACKUP CACHE 39

Introduction to the Disk Library Interface *39*

Overview of the Cache Layout *40*

Enabling Fast Cache Recovery *40*

Storage of Disk Library Backups *41*

DEFINING TAPE DRIVES 42

Use of Tape Device Groups *42*

Configuring Tape Library Devices *43*

Configuring Virtual Devices for MAC OSX *43*
Tape Block Size Considerations *43*
OVERVIEW OF BACKUP CLASSES, GROUPS AND MEDIA POOLS 44
Defining Backup Classes *44*
Defining Media Pools *45*
Media Definitions for Roaming Clients *46*
Example Media Pool Definitions *47*
Example 1: A Simple Two Level Tape Backup Strategy 47
Example 2: A Simple Two Level Disk Backup Strategy 47
Example 3: Two Level Tape Backups with Tape Library and Tape Mirroring for Offsite 47
Using Tags to Implement Multi-level Backup Strategies *48*
Example 4: Advanced Four Level Backups using Tags, Offsite Requirements, and Mixed Media Pools 48
Example 5: Special Case Two Level Backup with Tags 49
Defining Backup Groups *50*
BACKUP CONFIGURATION 51
MANAGING CLIENT BACKUP VOLUMES 51
Backup Group Definition File Format *51*
Adding New Client Volumes *51*
Default Backup Volumes *52*
Adding New Default Volumes *52*
Overriding Default Volume Definitions *52*
Managing Roaming Clients *53*
Server Disaster Recovery Configuration *53*
Windows® Registry Backup 53
Removing Client Definitions *53*
Authenticating New Backup Clients *54*
ANDREW FILE SYSTEM CONFIGURATION 55
Configuration for AFS .readonly Volumes *56*
Automating AFS® backups *56*
Multiple Cell Support *56*
AFS® Configuration Example *57*
USE OF RULE FILES 58
AUDITING CLIENTS 59
Sub-Directory Considerations for Auditing *59*
Network Auditing *59*
Running the Network Audit 60
Resolving Network Audit Errors 60
Alternative to Network Auditing 60
Class Definition Auditing *61*
Resolving Class Audit Errors 61
Automating Audit Processing *62*
Other Uses of the Audit Program *63*
Authenticating Clients 63
Debugging Clients 63
Listing Client Partition Information 63
Viewing/Updating Client Configuration Files 63
Viewing Client Revision Information 65
MEDIA MANAGEMENT 66
BACKUP CACHE MANAGEMENT 66

Automatic Cache Balancing ... *66*
Automatic Cache Clearing ... *66*
Manual Cache Relocation ... *66*
Configuration with the Disk Library Interface ... *66*
TAPE MANAGEMENT ... 67
Manually Labeling Tapes ... *67*
Viewing a Tape Label ... 67
Mounting Tapes ... *68*
Recycling Tapes ... *68*
Removing Tapes from the Tape Database ... *69*
Tape Library Commands ... *69*
Erasing Tapes ... *69*
Scanning Tapes ... *70*
BACKUP OPERATION ... 71
SERVER INITIATED BACKUPS ... 71
Full Network Backups ... *72*
True incremental Network Backups ... *72*
Cumulative Incremental Network Backups ... *72*
Synthetic Lower Level Backups ... *72*
CLIENT INITIATED BACKUPS ... 73
USE OF CLIENT SIDE PRE/POST PROCESSING SCRIPTS ... 74
AFS® BACKUPS ... 74
Updating Backup Volume Definitions ... *74*
Generating .backup Volumes ... *75*
Running Backups ... *75*
AFS® Volume Time Stamp Considerations ... 75
Backup of Additional Fileserver Directories ... 76
Backup of Additional Database Server Directories ... 76
Automating AFS® Backups ... 76
EXPIRING BACKUP VOLUMES FROM MEDIA POOLS ... 77
Individual Clients and Volumes ... *77*
Expiring Older Media Volumes ... *77*
Expiring Multiple Levels of Media Volumes ... *77*
Removing Older Media Volumes Permanently ... *77*
Expiring a Fixed Amount of Data ... *78*
Expiring Only a Disk Library Volume ... *78*
SCHEDULING BACKUPS ... 79
Scheduling Concepts ... *79*
Managing Media Pools ... *79*
Timing Considerations for Media Expiration ... 80
Automating Backups ... *80*
Example 1: TiBS Small Business Edition ... *81*
Example 2: TiBS Lite Version ... *82*
Example 3: Advanced Scheduling in TiBS Full Version ... *83*
Example 4: Production Script: mysiteauto ... *84*
RESTORE PROCEDURES ... 85
RESTORING TO AN ALTERNATE LOCATION ... 85
RESTORING OLDER BACKUP VOLUMES ... 85

INCREMENTAL RESTORES 85
SINGLE FILE OR SUB-DIRECTORY RESTORE 86
Limit the Search *86*
EXAMPLE FILESEARCH OUTPUT 87
WINDOWS® RECOVERY PROCEDURES 88
Recovering a System Drive *88*
Windows® Registry Edit 88
Booting the Recovered System 88
Recovering a Windows® Registry *89*
AFS® RESTORES 90
Restoring Incremental Data *90*
Restoring to an Alternate Location *90*
Restoring to a Specific Time *90*
Restore of AFS .readonly Volumes *91*
Disaster Recovery for AFS® *91*
Recovering an AFS® Fileserver 92
Recovering an AFS® Database Server 92

BACKUP MONITORING 93

MONITORING BACKUP SERVER ACTIVITY 93
BACKUP REPORT GENERATION 93
Filtering Reports *93*
Customizing Reports *93*
Cache Statistics *94*
Tape Statistics *94*

ERROR RECOVERY 95

LOCATION OF BACKUP SERVER LOG FILES 95
BACKUP SERVER FAILURES 96
BACKUP VOLUME ERRORS 103
LOCKED CACHE VOLUME RECOVERY 103
BACKUP CACHE RECOVERY 104
Cache Recovery Options *104*
Cache Recovery Failures *104*
Disk Library Recovery *105*
SERVER LOCK FILE RECOVERY 105
BACKUP SERVER RECOVERY 106
TAPE ERROR RECOVERY 107
Recovering a Lost Tape Database *107*
Removing Unreadable Tapes from the Tape Database *107*
For A Single Failed Restore Volume *107*
Recovering from a Failed Tape Merge *107*
Recover from previous tapes 107
Recover from the backup client 108

COMMAND REFERENCE 109

SUMMARY OF SERVER COMMANDS 109
AUTOMATION AND SERVER CONFIGURATION 110

BACKUP CLIENT CONFIGURATION 111
MEDIA MANAGEMENT 114
BACKUP OPERATIONS 117
RESTORE PROCEDURES 122
SERVER MONITORING 125
SERVER MAINTENANCE 127
SUPPORT PROGRAMS AND SCRIPTS 130
CLIENT COMMANDS 132
TECHNICAL NOTES 133
BACKUP AND RESTORE OF HARD LINKS 133
SERVER RESERVED CHARACTERS 133
SERVER RESERVED WORDS 133
RESTORE OF UNIX® SPECIAL FILES 133
MACINTOSH OSX RECOVERY NOTES 134
GLOSSARY OF TERMS 135

Welcome to TiBS 2.0

I would like to personally thank you for choosing the True incremental Backup System® (TiBS). Teradactyl® is committed to providing efficient and easy to use data management products. We developed the patented TeraMerge® technology to reduce the total network and client system load required for the backup function. TiBS will:

> ***Eliminate*** **periodic full network backup loads** on networks, file servers, workstations, and roaming clients by consolidating previous full backup data with incremental data to form new synthetic full backups (Full and Lite Versions only).
>
> ***Eliminate*** **periodic midlevel network backup loads** on networks, file servers, workstations, and roaming clients by merging previous incremental backup data to form new synthetic midlevel backups (Full Version only).
>
> ***Reduce*** **daily incremental network backup loads** on networks, file servers, workstations, and roaming clients by taking only True incremental backups over the network.

TiBS now comes with an optional disk library module to help manage backup server loads by reducing the size of incremental backup though our unique partial cumulative incremental backups!

This product is quite different from other backup systems. Our unique approach gives our customers the ability to re-think the way they run their backups. We recommend that you take the time to read and become familiar with the tools we have provided before settling on your long-term strategy. We are pleased to present you with the best backup technology available in the marketplace today.

Sincerely,

Kristen J. Webb
Partner and CTO
Teradactyl LLC.

Introduction

To take full advantage of the power of the True incremental Backup System® (TiBS), it is important to understand how it is different from other backup products you may have used or are currently using. Patented TeraMerge® technology gives you the capability to re-think how you do backups. Please take a moment to read the following sections. Once you understand all of the tradeoffs presented here, you will be ready to prepare your backup plan.

How Traditional Backup Systems Work

Most backup products use what is known as a "traditional level backup". The levels are usually defined from zero (0) to nine (9). The way backup levels work can be simple or complex depending on how they are scheduled. Level 0 is usually defined as the epoch or full backup. All of the data on a client volume is backed up during level 0 backup. All other level backups will copy all data changes since the most recent backup of any lower level. For example, a backup of level 2 sends changes since the last level 0 or level 1 backup (whichever is most recent).

In this traditional level backup scheme, there is a classic tradeoff between network and client loads, and the exposure to tape failure in the restore process. The more complex backup schedules offer significantly reduced network and client loads but require more tapes to restore data completely. The need for more tapes in the restore process increases the exposure to an individual tape failure. A single bad tape can lead to an inability to recover lost data. Fewer levels can improve restore time and reliability but the cost is increased loads on networks and clients.

Most of our customers use 2 to as many as 5 backup levels. This seems to be the optimal range where backup administrators are most comfortable with the classic tradeoff limitation of a traditional level backup. Level 0 backups are run weekly, monthly, quarterly, semi-annually, or annually depending on the amount of data, the available network bandwidth, the available backup window, and the perceived cost associated with loss of data. Sites that value their data more and can afford it tend toward 2 or 3 backup levels. Sites that need to minimize the impact of backups for any reason tend toward 4 or more backup levels.

How TeraMerge® Works

TeraMerge® technology works on the fundamental concept of data reuse. A TiBS client only sends file changes that have occurred since the last network backup (full or incremental). We refer to this as the “True incremental Backup”. The TeraMerge® process then uses the current data changes with previous backup information, reusing data that has not changed since the last backup to generate all remaining backups synthetically.

With TiBS Small Business Edition, administrators can curb the impact of nightly incremental backups by capturing True incremental data from backup clients and integrating that information with data in the backup server disk cache to produce a synthetic cumulative incremental backup. Additional savings in network workloads can be achieved through the use of our synthetic partial cumulative incremental backups that can extend the time between each network full backup.

TiBS Lite eliminates the need to take periodic network full backups. The TeraMerge synthetic full backup consolidation process produces new full backups that contain the same data as those that would have been produced by retaking data over the network.

The Full version of TiBS extends the idea of data reuse even further. The TeraMerge® multi-level backup consolidation process merges True incremental data gathered from the client with previous data on disk or tape to produce a new synthetic full, cumulative incremental, or partial cumulative incremental backup. Each new backup volume contains the same data that would have been taken from the backup client at that time. These backups are generated with absolutely no interaction with the backup clients or the network. This allows larger sites to reduce the backup server resources required to maintain backups, allowing the same backup hardware to support larger amounts of data.

New Disk Library Interface (DLI)

The TiBS backup cache is used to store current incremental data and is used as a staging area for the generation of new lower level backups. All versions of TiBS now have the ability to extend the caching of backups on disk to the storage any backup to disk. The DLI allows TiBS to optionally run as a Disk-to-Disk backup solution. When, combined with tape storage, the DLI provides for more efficient backup and restore processing, allowing the same backup hardware to support increased amounts of data.

Types of Backup Available with TiBS:

Network Full	All file data including meta data such as directories, folders, filenames, file attributes, access control lists and other security information is taken from the backup client over the network. This type of backup is available in all TiBS versions.
Network True Incremental	All directory and meta data information is taken from the backup client over the network. Only those files that have been modified or created since the last backup are sent from the client. This type of backup is available in all TiBS versions. The Disk Library Interface option is required for use with TiBS Full and Lite versions.
Network Synthetic Cumulative Incremental	All directory and meta data information is taken from the backup client over the network. Only those files that have changed since the last backup are sent from the client. The server then consolidates previous incremental backup data, omitting obsolete or deleted files, to generate a new synthetic cumulative incremental backup volume. This type of backup is available in all TiBS versions.
Synthetic Partial Cumulative Incremental	Current data in the backup cache is consolidated with a previous higher level incremental backup data on disk to produce a new synthetic backup at this level. This feature may be used to implement multiple levels of incremental backups. This option is only available with the Full Version.
Synthetic Cumulative Incremental	Current data in the backup cache is consolidated with a previous cumulative incremental backup at this level to produce a new synthetic backup. This feature may be used to implement multiple levels of incremental backups. This option is only available with the Full Version.
Synthetic Full	Current differential data in the backup cache is merged with the most recent full backup to produce a current synthetic full backup. This option is only available in TiBS Full and Lite Versions.

Redundancy Considerations in TiBS

The TeraMerge synthetic backup consolidation process with TiBS provides built in data redundancy. Each new synthetic backup is the product of one ore more previous backups. If a synthetic backup volume fails, data can be retrieved from the backup volumes used to produce it. This redundancy does depend on careful backup volume retention policies. Note that once older backup volumes have been deleted their contribution to data redundancy will be lost.

Use of the Disk Library Interface can provide additional levels of protection when used in conjunction with tape.

The redundancy in TiBS Small Business Edition applies to *the last successful Cumulative Incremental backup*. TiBS Small Business Edition does not maintain redundancy of full backup volumes.

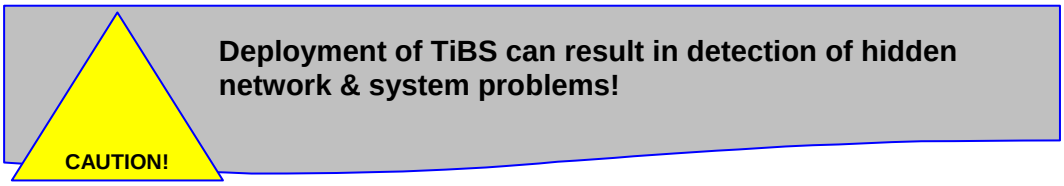
Use of tape mirroring can provide additional protection for all backups. Tape mirroring is also recommended for offsite management of backup data.

Conventions Used in the Document

The following fonts are used throughout this document:

Arial Bold – Blue	**Chapter and section headings**
Times New Roman	General Text
`Courier New`	`To distinguish commands, command line output, and configuration files from ordinary text`
Arial Bold	**Notes, tips, and cautions**
Italic font	*Emphasis for something important*
Bold font	**Emphasis for something new and important**

Note: **Technical Support E-Mail: support@teradactyl.com**



TIP: **Proper feeding of your pterodactyl can extend the life expectancy of your other pets.**

Example configuration file:

```
# comments
data
```

Manual Overview

This document is a comprehensive resource for installing, configuring, and managing the TiBS products. Below is a summary of the major sections of the manual:

Introduction	Initial setup of the TiBS client and server software.
Server Configuration	Backup server hardware configuration.
Backup Configuration	Configuration of backup clients and auditing procedures.
Media Management	Disk drive, tape device, and removable media management.
Backup Operation	Common commands and options available for backup processing.
Restore Procedures	Common commands and options available for restore processing.
Backup Monitoring	Reporting and statistics to track and optimize backup processing.
Error Recovery	Error recovery procedures.
Command Reference	Complete list of programs and their available options.
Technical Notes	Notes that describe how TiBS performs under certain conditions.
Glossary of Terms	Principal terms and definitions.
Index	Index for easy location of subject matter.

Installation

Pre-Install Requirements

Before installing the TiBS server, be prepared with the following:

The installation destination: This should be on a disk partition with plenty of space. Plan on 1% of the total data size you expect to backup at a minimum. For example, if you plan to backup a terabyte a 10-gigabyte hard drive would do. Mount this drive partition onto **`/usr/tibs`** or the directory where you plan to install the server software. The installation directory does not need to be a mount point, but there should be enough free space on the partition where the server is to be installed.

Backup cache requirements: One or more disk partitions make up the backup cache. All data written to tape goes through the cache by default. Incremental data remains in the cache. Full data remains in the cache until written to tape. The amount of cache space required can vary greatly from site to site. We recommend starting out with at least 10% of the total size of the data you expect to backup at a minimum. For example, if you plan to backup three terabytes of data, the initial size of the cache should be about 300 gigabytes. The cache can be spread over multiple cache partitions. For example, a 300-gigabyte cache could be implemented with 5 – 70 gigabyte or 2 – 180 gigabyte disk partitions.

Install Options

There are several command line options available to customize the installation of the TiBS backup server. Review these options and select the options that are appropriate for your site before running the server install.

`-s`	Perform a TiBS server install **(required)**
`-u`	Perform an upgrade. Only updates binaries and scripts.
`-a path`	Perform install from an alternate CD image. Used to correct problems with install on a CD.
`-d domain`	Specify the server's domain from the command line. If the install cannot figure out the fully qualified hostname the user is prompted later.
`-p path`	Location on the server to install TiBS. The default path is `/usr/tibs/.`
`-P port`	Install an `inetd` service that gives access to remote clients at an alternate TCP/IP port. The default port is 1968.
`-R release`	Use an alternate OS release if the server's full release number is not found on the CD.
`-T tcpd`	Install the `inetd` service using the path to `tcpd` for added network security. Remote access to the backup server may then be controlled with `/etc/hosts.allow` and `/etc/hosts.deny`.

Server Installation

Mount the install media onto a CD-ROM reader on the backup server and run as root with the **-s** flag:

```
# /cdrom/install.sh -s <additional_install_options>
```

The installation program will display the terms of the license agreement and the following:

```
To accept these terms, type "I agree": I agree

Enter the install path for TiBS server [/usr/tibs]: /usr/mypath

TiBS uses fully qualified hostnames for configuration and security.
Enter the full hostname of this server [host]: host.domain

Installing the TiBS server

FROM: ./release/arch/os/revision
TO:   /usr/mypath
```

If your backup server is running a firewall and you want to allow backup clients to initiate their own backups, you need to specify a rule for the `tibsd` program at the configured TCP/IP port (1968 by default).

NOTE: A reboot may be required for TiBS OSX Panther servers to enable `xinetd` if no other `xinetd` services are currently configured.

RLM License File

In order to run backups using TiBS you must obtain a valid license file from Teradactyl®. You will need the fully qualified `hostname` and `hostid` of the backup server. You can determine the `hostid` with the utility `rlmutil`:

```
# /usr/tibs/rlm/rlmutil rlmhostid (for Solaris)
# /usr/tibs/rlm/rlmutil rlmhostid –ether (for Linux)
```

You can request a new key on line from the Teradactyl® Customer Center at:

http://www.teradactyl.com/Customers/KeyForm.html

Place this file in the **`rlm`** directory as **`tibs.lic`** (or on an appropriate license server) before starting TiBS. If you are updating your license, you must restart the license server for the new key to take effect:

```
# /usr/tibs/rlm/rlmstop
# /usr/tibs/rlm/rlmstart
```

If you backup server is running a firewall, you need to specify the license manager TCP/IP ports in the `rlm.dat` file by modifying the **`HOST`** (e.g. port 2764) and **`ISV`** (e.g. port 5000) lines:

```
HOST tibsosx.nm.teradactyl.com 000393da9f5c 2764
ISV tibsrlm /usr/tibs/rlm/tibsrlm 5000
```

You can then configure the firewall to allow the `rlm` and `tibsrlm` programs to read from these ports.

Uninstalling the TiBS Server

To uninstall the TiBS server, run the following from the install media:

```
# uninst.sh -s
```

Client Installation for UNIX®

Client Install Options

There are several command line options available to customize the installation of the TiBS backup client. Review these options and select the options that are appropriate for your site before running client installs.

-u	Performs an upgrade. Only updates client binaries.
-a path	Performs install from an alternate CD image. Used to correct problems with install on a CD.
-d domain	Specifies the client's domain from the command line. If the install cannot figure out the fully qualified hostname the user is prompted later.
-n server	The fully qualified hostname of the authentication (primary backup) server.
-p path	Location on the client to install TiBS. The default path is `/usr/tibs/client`.
-P port	Installs the client service at an alternate TCP/IP port. The default port is 1967.
-R release	Uses an alternate OS release if the client's full release number is not found on the CD.
-T tcpd	Installs the client's `inetd` service using the path to `tcpd` for added network security. Remote access to the backup client may then be controlled with `/etc/hosts.allow` and `/etc/hosts.deny`.

Mount the install media onto a CD-ROM reader on the backup client and run as root:

```
# /cdrom/install.sh <install_options>
```

The installation program will display the terms of the license agreement and the following:

```
To accept these terms, type, "I agree": I agree

Enter the install path for TiBS client [/usr/tibs]: /usr/mypath

TiBS uses fully qualified hostnames for configuration and security.
Enter the full hostname of this client [client]: client.domain

Enter the full hostname of the server [none]: server.domain

Installing the TiBS client

FROM: ./release/arch/os/revision
TO:   /usr/mypath
```

If your backup client is running a firewall, and you want to allow the backup server to communicate with the backup client automatically, you need to specify a rule for the `terad` (`terad.exe` for Windows) program at the configured TCP/IP port (1967 by default).

NOTE: A reboot may be required for OSX Panther clients to enable `xinetd` if no other `xinetd` services are currently configured.

Installing UNIX® Clients from the Backup Server

Installations for UNIX® clients may be performed directly from a backup server that has `rsh` or `ssh` support for the root user to the clients. In addition to the options listed above for client install, there are install options for managing client installations from the backup server.

`-C`	Perform a remote client upgrade on all hosts defined in `state/clients.txt`. Do not use this feature if you are running Windows® clients.
`-c client`	Perform a remote client install on the specified host. The hostname should be the fully qualified hostname (`host.domain`) of the backup client.
`-f filename`	Perform multiple client installs from the list specified in `filename`. The format of the file is a single fully qualified hostname per line. Any UNIX® clients may be included in the list. The install will determine individual OS requirements on each host.
`-S shell`	Use an alternate shell to access clients during install. The default shell is `rsh`.

Installing a Single Client

A single client may be installed using the `-c` option to `install.sh`:

```
# /cdrom/install.sh -c host.domain <install_options>
```

Installing Multiple Clients

Multiple clients may be installed using the **`-f`** option to `install.sh`:

```
# /cdrom/install.sh -f filename <install_options>
```

The file contains a list of all of the clients (one per line) that are to be installed.

Example installation file:

```
host1.domain
host2.domain
```

Uninstalling UNIX® Clients

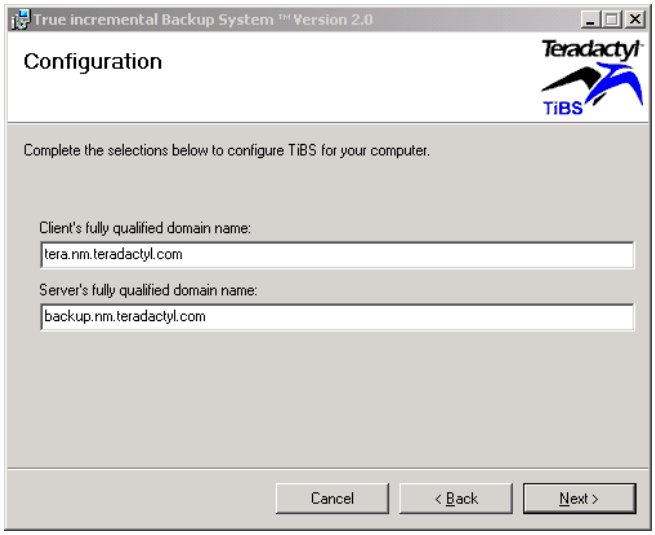
From the directory where the TiBS client was installed, run the following from the install media:

```
# uninst.sh
```

Installing Windows® Clients

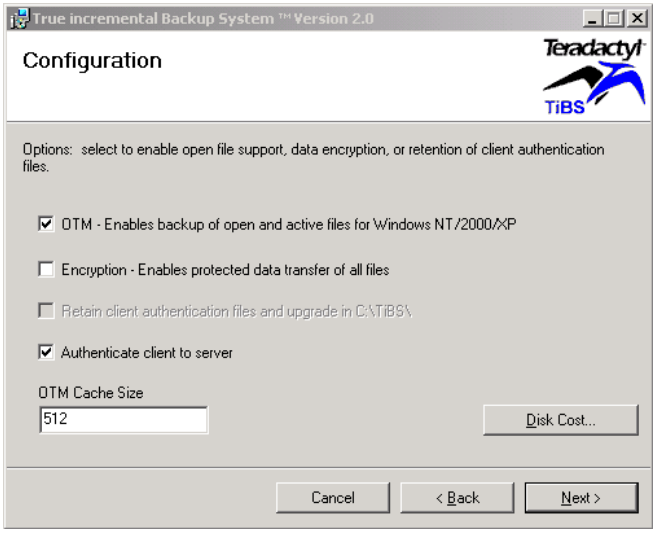
The TiBS install media now ships with a Windows installer Graphical User Interface (GUI). The install program should run automatically whenever the TiBS CDROM is mounted on a Windows machine. If the installer does not start automatically, or if you are installing from a network location, click on **`setup.exe`**.

1. At the splash screen click "**Next**".
2. At the welcome screen click "**Next**".
3. Read the license agreement carefully and click **"I Agree"** to continue with the install.
4. At the first configuration screen, enter the fully qualified hostname of this backup client and the fully qualified hostname of the backup server that the backup data will be sent to. If the backup client does not have a static IP address, it may not have a fully qualified hostname. In this case, you may use an arbitrary unique hostname to identify the backup client to the backup server (see managing roaming clients).



5. Click "**Next**" to go to the next configuration screen.
6. If you purchased an OTM license from Teradactyl®, select the OTM option. This will reveal an additional option. The OTM Cache Size should be a minimum of 1% of the largest hard drive you intend to backup on this system. For example, if you have a 20 Gigabyte drive you intend to backup, you will need 200 megabytes of OTM cache at a minimum. Use the "**Disk Cost**" tab to review the partition sizes for this host.
7. If you want to ensure that the transfer of data to and from this client (through the TiBS backup and recovery services) is protected over the network, select the Encryption option. Enabling encryption will slow backup performance significantly. The encryption option, should only be used when necessary.

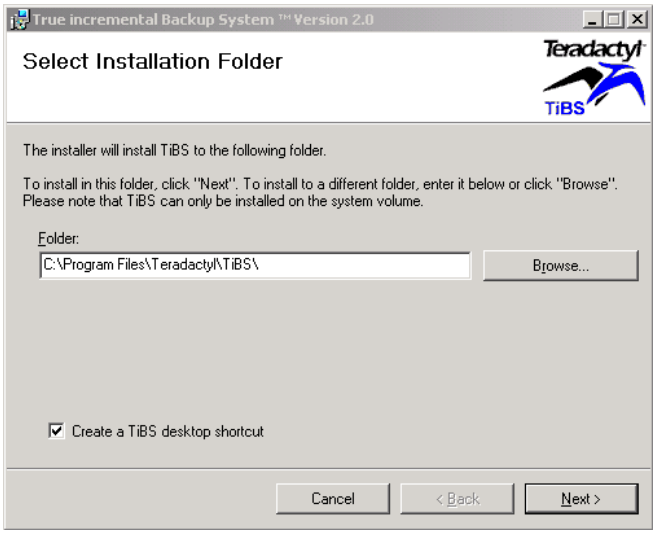
8. If this is a new install and the backup client is able to connect to the backup server, select "**Authenticate client to server**" to register the backup client with the backup server. If TiBS has already been installed on this host, you may select "**Retain client authentication files**" to keep the current registration information.


True incremental Backup System ™ Version 2.0
Configuration
Teradactyl
TiBS
Options: select to enable open file support, data encryption, or retention of client authentication files.
OTM - Enables backup of open and active files for Windows NT/2000/XP
Encryption - Enables protected data transfer of all files
Retain client authentication files and upgrade in C:\TiBS\
Authenticate client to server
OTM Cache Size
512
Disk Cost...
Cancel
< Back
Next >

9. Click "**Next**" to go to the next configuration screen.

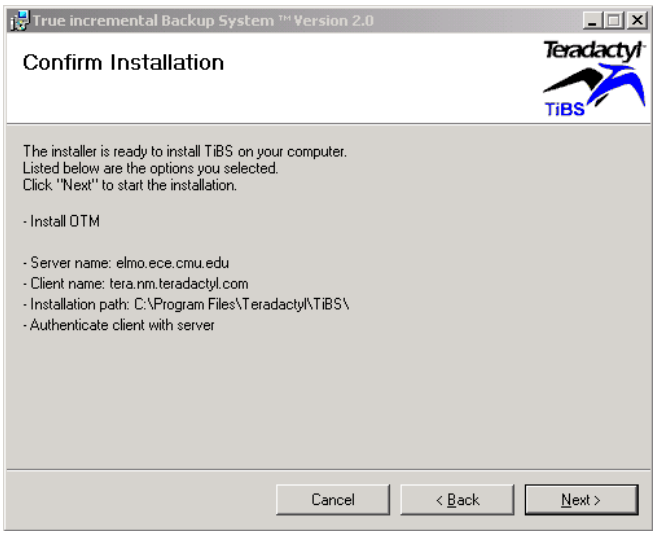
10. If TiBS was previously installed, the current location is displayed; otherwise the default location is displayed. To choose an alternate folder for the TiBS client, enter the name in the "Folder" window or by using the Browse option. **The TiBS client for Windows® must be installed on the system drive.**

11. If the user wishes or needs to initiate backups on his own, for example, a DHCP laptop that does not have a static IP address, select the "**Create a TiBS desktop shortcut**".

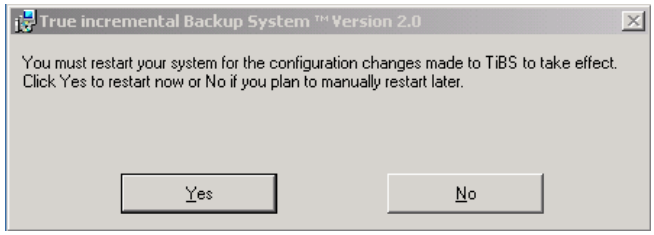

True incremental Backup System ™ Version 2.0
Select Installation Folder
Teradactyl
TiBS
The installer will install TiBS to the following folder.
To install in this folder, click "Next". To install to a different folder, enter it below or click "Browse".
Please note that TiBS can only be installed on the system volume.
Folder:
C:\Program Files\Teradactyl\TiBS\
Browse...
Create a TiBS desktop shortcut
Cancel
< Back
Next >

12. Click "**Next**" to go on to the Confirmation screen.

13. Review your selection options and settings.


True incremental Backup System ™ Version 2.0
Confirm Installation
Teradactyl
TiBS
The installer is ready to install TiBS on your computer.
Listed below are the options you selected.
Click "Next" to start the installation.
- Install OTM
- Server name: elmo.ece.cmu.edu
- Client name: tera.nm.teradactyl.com
- Installation path: C:\Program Files\Teradactyl\TiBS\
- Authenticate client with server
Cancel
< Back
Next >

14. Click "**Next**" to proceed with the installation.
15. Once the installation completes you will see a final installation screen confirming this. Click "**Close**" to finish the installation.
16. If you have chosen the OTM option, the backup client will require a system reboot and you will see the following screen once the installation is complete:


True incremental Backup System ™ Version 2.0
You must restart your system for the configuration changes made to TiBS to take effect.
Click Yes to restart now or No if you plan to manually restart later.
Yes
No

Upgrading/Uninstalling Windows® Clients

To upgrade or uninstall a Windows® backup client, run the `setup.exe` program from the install media or from the network location at your site where the install image for Windows® is located. Choose the "Repair/Update" option to update the TiBS client. Choose the "Remove" option to uninstall TiBS. Click on "Next" to finish.

Installing TiBS via Group Policy

TiBS can be silently installed to multiple PCs by taking advantage of Microsoft's Group Policy and IntelliMirror features provided with Active Directory. Three steps are involved:

1. Copy the TiBS installation media to a network share accessible by all target PCs;
2. Customize the `tibs.msi` file to reflect site defaults for TiBS configurations;
3. Create a TiBS installation policy.

The only tool required in addition to Active Directory is an MSI table editor. Microsoft's Orca has been provided in the `msitools` directory on the TiBS installation CD, but any tool capable of producing MST transforms should be acceptable.

Copy the TiBS installation media to a network share

Most likely, an appropriate network share is already available if Group Policy is being used to distribute software. If not, simply create a share and ensure that both share-level and file system-level permissions are appropriate. Copy the `tibs.msi` file and the `\release\i386\Windows` folder from the installation CD to the appropriate share. Note that the only required files are `tibs.msi` and the release folder. All other files and directories on the installation media can be excluded from the deployment share.

Customize tibs.msi with site defaults

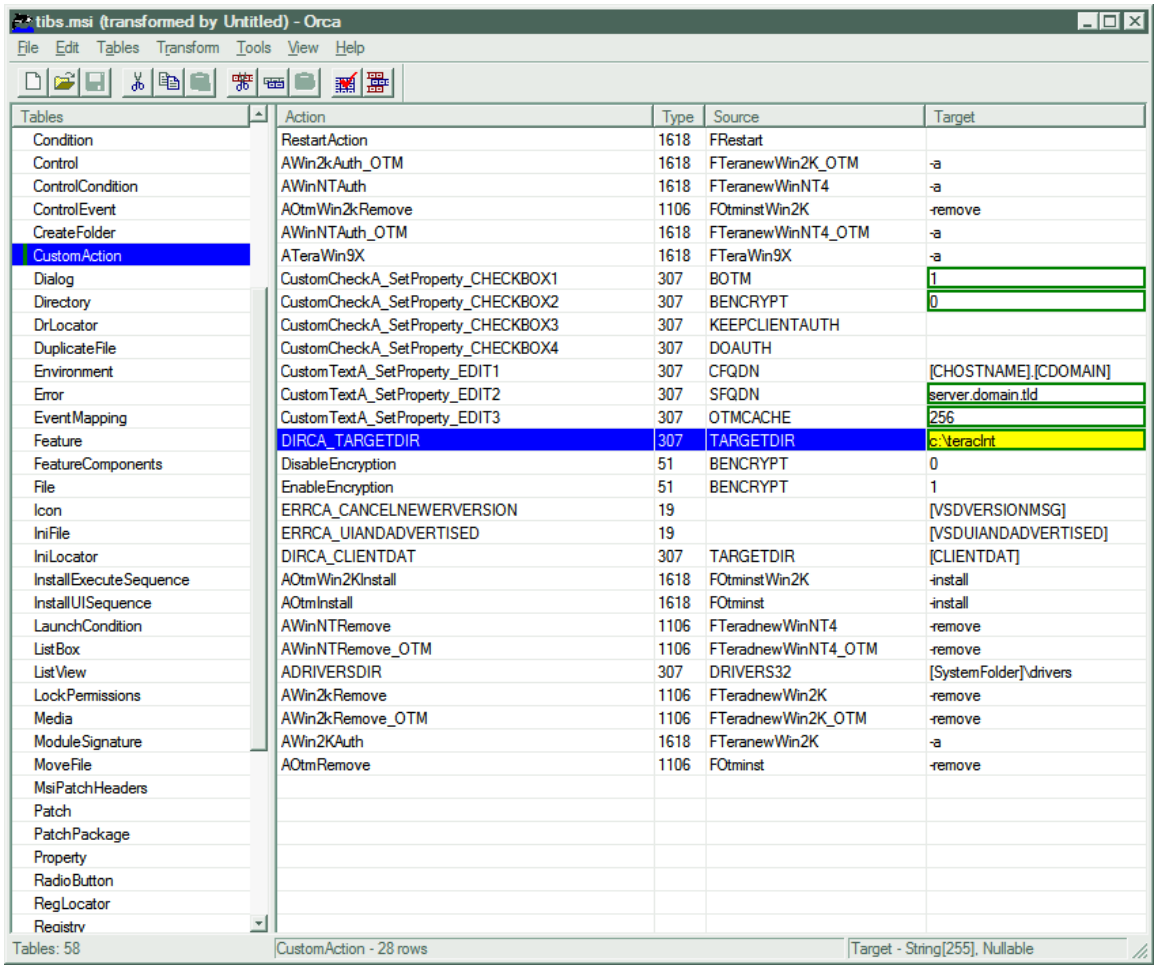
Any TiBS configuration option available in the GUI installer can be set via an MST transform applied to the `tibs.msi` file. Each option is set in the `tibs.msi` CustomAction table through the following properties.

`BOTM=1|0`	OTM installed|not installed
`BENCRYPT=1|0`	Encryption enabled|disabled
`SFQDN=string`	TiBS server host name
`OTMCACHE=integer`	OTM cache size
`TARGETDIR=string`	TiBS installation directory

The client host name is set by the installer from existing values in the client's registry. If this default is unreasonable, the CFQDN property can be set.

Using Orca, an MST file can be created. To generate an MST file in Orca, open `tibs.msi`, select Transform→New Transform, change the Target column for any of the Source columns listed above, and then choose Transform→Generate Transform to save the MST file. The MST file should be copied to the network deployment share that holds `tibs.msi` and the release directory.

Setting Target values to generate an MST



Create TiBS installation policy

TiBS must be assigned to a machine account through a Group Policy Object's Computer Configuration properties. TiBS should not be published to a user account since it requires system-wide changes and provides functionality independent of any user account. To assign TiBS, edit the Computer Configuration Software Installation settings of the appropriate Group Policy Object and add a new package assigning TiBS with the MST file prepared above as the package modification. The 'advanced' option will need to be chosen when creating the new installation policy so that the 'modification' setting can be changed.

When client PCs apply the policy during reboot, they will apply the MST transform to `tibs.msi` and run the install from the deployment network share. Defaults in the installer coupled with values provided by the MST will ensure that each client has TiBS installed, authenticated, and configured.

Installing Windows® Clients Remotely

TiBS provides a script that enables a Windows® client to be installed from a central Windows workstation or server. You can use these scripts to install a single client or several clients at once. The required arguments will vary depending on the type of installation that is performed.

```
D:\> wininst [required arguments] [options]
```

Required arguments for different installations:

	NT	2000 or XP
Os identifier	`/n`	`/2`
Fully Qualified Server	`/s server.domain`	`/s server.domain`

One of the following options is required:

Install Remote Client	`/c client.domain`
Install Multiple Remote	`/f filename.txt`

Additional options may be added:

Install CDP's OTM	`/o` (Reboot Required)
Specify OTM cache size	`/oc megabytes`
Reboot Client	`/r`
Upgrade	`/u`
Alternate Install Path	`/p path`
(default=`\Progra~1\Teradactyl\TiBS`)	**TiBS must be installed on the system drive**

Example installation file:

```
host1.domain
host2.domain
```

For example, to install a remote Windows® NT system with Columbia Data Product's Open Transaction Manager™, and make it active (This includes a system reboot):

```
D:\> wininst /n /s server.domain /c client.domain /o /r
```

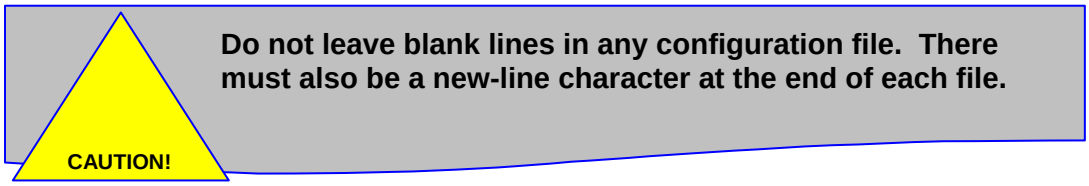
To remotely install multiple NT clients from a single location run:

```
D:\> wininst /n /s server.domain /f filename.txt
```

Note: In order to install remote clients, you must have the Scheduler service running on the target computer. You can enable Scheduler and other services with `srvmgr.exe`.

Server Configuration

All server configuration files are located in the **`state`** directory. The files described in this section may be modified using any text editor, such as **`vi`** or **`emacs`**.



All configuration files use the special character '`|`' as a field separator. This character is reserved and cannot be part of the value of any field. All configuration files support the comment character '`#`' at the beginning of any line. Comments are not supported part way through an actual definition line.

Example configuration file:

```
# this is a valid comment
this is # not a valid comment
value1|value2|value3
```

Starting and Stopping the Server

You may manually update most server configuration files during periods of server inactivity; however we recommend shutting down the server with **`stoptibs`** before making any updates:

```
# stoptibs
```

Once all configuration changes are completed, restart the server with **`runtibs`**:

```
# runtibs
```

Four processes should always be running on a TiBS backup server. They are:

cachemgr:	Monitor backup cache and take corrective action to prevent cache overflows.
`rlm:`	The Reprise License Manager™ (RLM).
`tapemgr:`	Initialize defined tape drives at boot time. Monitor backup activity and handle tape and disk library write requests.
`tibsrlm:`	The Reprise License Manager™ (RLM) daemon for TiBS is required to run backups.
`tibswd:`	Monitor the `cachemgr`, `tapemgr`, and `teramgrd` processes. If any of these processes die, `tibswd` will start another process.

One way to start TiBS is by editing the **`/etc/inittab`** file with the `runtibs` command:

```
tibs:5:once:/install_bin/runtibs > /dev/console 2>&1
```

Configuration Files Overview

Configure the following files during initial installation. Refer to the sections below for individual file syntax:

tibs.conf:	Backup server site configuration file.
caches.txt:	Hard disk definitions for the backup cache.
drives.txt:	Tape device definitions.
classes.txt:	All backup class definitions.
labels.txt:	Media pool definitions for each class.
groups.txt:	Backup group definitions for each class.
ThisFull.txt	Current full backup media pool for each class.
ThisDaily.txt	Current incremental backup media pool for each class.
clients.txt:	Network audit backup client definition file.
subnets.txt	Network audit subnet definition file.

Configuration of tibs.conf

This is a shell resource file and is not subject to the rules for editing configuration files described above. Current configuration parameters and their functions are:

Parameter	Defaults	Description
TIBS_HOME	/usr/tibs	The product install directory for the TiBS server. Defined at install time and should not be changed unless this directory is re-located.
MY_HOST_NAME	server.domain	The fully qualified hostname of this backup server.
TIBS_ATLI_ENABLE	1	Enable tape operations to use the automated tape library interface (ATLI).
TIBS_ATLI_INITIALIZE	1	Set to 1 to enable all library commands to send a SCSI initialize request. Some older libraries do not support this functionality. Set to 0 to disable this feature.
TIBS_LIBRARY_COUNT	1	Total number of tape library devices configured for this server, /dev/atli* (/vdev/atli* for Panther), TiBS Small Business Edition supports a maximum of 1, TiBS Lite supports a maximum of 2, and the Full Version supports any number of library devices.
atli_dbd	0	Set to the value of 8 if your tape library supports the return of one ore more block descriptors in the SCSI MODE SENSE command. Set to 0 otherwise. For more information, please consult your tape library manual or contact Teradactyl for additional assistance.
TIBS_ATLI_DEBUG_LEVEL	0	This setting is used to debug the behavior of SCSI commands for medium changers during various library state changes, such as an import/export door open, or a library power cycle. This value should normally be set to 0.
TERA_ADMINS	""	Space separated list of e-mail address to notify for alerts and reports. We recommend that more than one person be defined.
REPORT_ADMINS	""	Optional space separated list of e-mail address to receive reports for personnel who need to view summary information on backup progress, but do not participate in day-to-day backup operations.
MAX_REPORT_SIZE	1024	The maximum size in kilobytes that e-mailed reports may be. If a report's size is greater that this, a warning message is sent instead. The oversize report can be obtained directly from the backup server.

Parameter	Defaults	Description
TERA_NOTIFY_COMMAND	teranotify	The name of the script or program that is called whenever notifications or alerts are sent from TiBS programs. The default script sends an e-mail message to all defined TERA_ADMINS.
TERA_LOCAL_SCRIPTS	""	A listing of programs and scripts (e.g. mysitescript) that are written locally, which call TiBS programs and scripts. These are the first processes that stoptibs will shutdown before stopping other TiBS programs and scripts.
TERA_CUSTOM_LOGS	""	Site specific output files located in ${TIBS_HOME}/reports. Each output file (e.g. customlog.txt) will be included by genreport and filtered by the file customlog.flt.
TERAWATCH_SLEEP	300	Time in seconds that tibswd will sleep between checks. Default is 5 minutes.
TERAWATCH_LOCK	12	The tibswd daemon will wait a specified number of iterations for the current lock file to be removed by a running process before reporting or removing it.
TERAWATCH_CLEAR	0	Set value to 1 to permit the tibswd to automatically remove hung lock files that have been detected.
TAPEMGR_SLEEP	120	Time in seconds that the tapemgr daemon sleeps between checks for outstanding tape mount requests. Default is 2 minutes.
TAPEMGR_NOTIFY	5	The frequency of e-mail alerts is computed as a product of this variable and TAPEMGR_SLEEP. The default frequency of e-mail alerts is 10 minutes.
TIBSTAPED_SLEEP	120	Time in seconds that the tape daemon processes wait between checks for pending backup jobs. Jobs are continually processed until none are available before a tape daemon sleeps again.
TIBSTAPED_VERIFY_CACHE_DATA	2	Used to signal the tibstaped processes to scan the cache volume for checksum errors before writing data to tape. Possible values are: 0: do not verify cache volumes 1: verify cache volumes that will be cleared from the cache 2: verify all cache volumes
TIBSTAPED_ARCHIVE_PARTCUM	1	Set to 1 to enable tibstaped to automatically archive processed partial cumulative incremental backups. Sites with more complexity (tens of thousands of backup volumes) should set this value to 0 and then use dlremove periodically to archive these olumes instead.

Parameter	Defaults	Description
TIBSTAPED_PROCESS_PENDING	25	Set to 0 to allow each tibstaped to process ny new pending backup jobs. For larger sites with multiple tape devices this value controls how many volumes from the incoming queue each tibstaped will process at a time. A typical useful value may range from 10 to 1000.
TIBSTAPED_POSITION_CHECKS	0	Track offset progress on tibstaped, only needed in rare cases where tapes are being overwritten (1==enable, 0==disable)
TIBSMNT_ALERT_WRITE	1	Email alerts when tape needed for write not found in library (1==enable, 0==disable)
TIBSMNT_ALERT_DRIVE_COUNT	1	Email alert when there are not enough drives available for a mount request (1==enable, 0==disable)
TIBSMNT_ENABLE_AUTO_RECYCLE	0	Set to 1 to enable or 0 to disable. When enabled, allows the tibsmnt program to recycle empty tapes from existing tape pools when there are now more blank tapes available in the appropriate tape library.
TIBS_TAPE_MAX_REUSE	0	Number of times that a tape may be recycled before it is no longer available for write requests. The default value of zero indicates that a tape may be used indefinitely.
DEFAULT_TAPE_BLOCKSIZE	256	The default tape block size in kilobytes. The valid range is from 16 to 256. Larger tape block sizes may create tapes that are not readable on other operating systems but may improve tape read and write performance dramatically.
TIBS_TAPE_CLEANER	<BLANK>	Set to a unique string that can identify tape cleaning cartridges. Leave blank if automatic tape cleaning is not desired. The string can be part of a barcode label (e.g. CLNS) or the name of the pool used for cleaning tapes (e.g. Cleaner).
TIBS_CLEANER_REUSE	0	Maximum number of times a tape cleaning cartridge should be used. Check the manufacturer's recommendations for this value.
USE_TAPE_COMPRESSION	1	Used by Linux backup servers only. The value of this setting is determined by the version of the UNIX `mt` command running on your backup server. If the `mt` command supports the `compression` option, set to 1 to enable tape compression or 0 to write data uncompressed to tape. . If the `mt` command supports the `datcompression` option, set to 3 to enable tape compression or 2 to write data uncompressed to tape. Refer to the UNIX man page for `mt` to determine the value for this setting.

Parameter	Defaults	Description
TIBS_TAPE_UNLOCK	1	Used by Linux backup servers only. Some tape devices require the `mt` unlock command to be issued before a tape can be ejected from a tape drive. Set the value to 1 to enable the `mt` unlock command or 0 to disable. If you are unsure, use the default value of 1.
CACHEMGR_SLEEP	120	Time in seconds that the `cachemgr` daemon waits between checks for cache overflows. During an overflow the `cachemgr` will not sleep again until all overflows have been resolved or reported.
CACHEMGR_NOTIFY	5	The frequency of e-mail alerts is computed as a product of this variable and `CACHEMGR_SLEEP`. The default frequency of e-mail alerts is 10 minutes.
CACHEMGR_BALANCE	0	Enable the `cachemgr` to resolve potential cache overflows by automatically relocating volumes to another portion of the backup cache (see Automatic Cache Balancing).
CACHEMGR_MAX_BALANCE_SIZE	500	Determine the maximum size of a cache volume (in megabytes) that the `cachemgr` will move to free needed cache space on a cache partition.
CACHEMGR_MIN_BALANCE_SIZE	5	Determine the minimum size of a cache volume (in megabytes) that the `cachemgr` will move to free needed cache space on a cache partition.
CACHEMGR_CLEAR	0	Enable the `cachemgr` to resolve potential cache overflows by automatically writing volumes to tape (see Automatic Cache Clearing).
CACHE_FAST_RECOVERY_DIR	/usr/tibs/recovery	Optional directory to place fast cache recovery information. This information allows failed portions of the backup cache to be recovered quickly, without having to load tapes. This feature may also be used to recover the cache in the event of a merge tape failure by re-synchronizing the backup client over the network. Check the size requirements for your site before enabling this feature.
CACHEMGR_CLEARLOCKS	0	Set to 1 to enable the `cachemgr` program to automatically clear old locks to the backup cache. Set to 0 to disable this feature.
CACHEMGR_TIBS_DIR_LIMIT	80	Percentage (1-100) utilization of the partition that the `${TIBS_HOME}` directory is on. If this value is met or exceeded, an alert message is generated.
CACHEMGR_RECOVERY_DIR_LIMIT	80	Percentage (1-100) utilization of the partition that the `${CACHE_FAST_RECOVERY_DIR}` directory is on. If this value is met or exceeded, an alert message is generated.

Parameter	Defaults	Description
ENABLE_FULL_BACKUPS	0	Set to 0 for TiBS or TiBS Lite and to 1 for TiBS Small Business Edition. When set to 0 this prevents clients from running a network full backup more than once.
ENABLE_INCREMENTAL_BACKUPS	1	Set to 0 to disable incremental network backup processing or 1 to enable incremental network backup processing. Typically used by sites that wish to temporarily disable incremental processing when running a large full backup job.
ALWAYS_INCREMENTAL_BACKUP	1	Set to 1 to enable roaming clients to initiate incremental network backups on demand, not just once a day or the set maximum scheduled. This is useful for sites that have roaming clients such as laptops that the server can attempt to backup once each day, but the user can request a backup at any time (e.g. before leaving the site). Set to 0 to disable.
ENABLE_MERGE_BACKUPS	1	Set to 0 to disable tape merge processing. Set to 1 to enable tape merge processing.
TIBS_MERGE_FROM_TAPE	1	Set to 1 to force backup consolidation at the current level to occur from tape. This is allows the `teramerge` process to verify tapes over time. Set to 0 to allow consolidations at the same level to occur from disk by default, and from tape only if necessary.
TERAMERGE_VERIFY_CACHE_DATA	1	Used by teramerge to perform checksum analysis on data in the backup cache that will beincluded with the current backup volume being generated (1=enable, 0=disable).
TIBS_FAIL_ON_EMPTY_DIRECTORY	1	Used to define the behavior of empty directory backups. Valid settings are: 0: Do not fail when the backup of an empty directory is taken 1: Warn if empty directory is taken, but do not fail 2: Fail and log error if an empty directory backup is attempted
NETWORK_BACKUP_WAIT	60	Time in seconds between requests for single threaded backup clients. Current singled threaded clients included Windows and Mac OS9.
WATCHDOG_TIMEOUT_WAIT	0	Set to 1 to enable or 0 to disable. When enabled, network backup processes will disable the network monitor before entering the final clean phases of a backup. Set to 1 only if your server is reporting server file access errors during network backups.

Parameter	Defaults	Description
TIBS_UPGRADE_SCRIPT	<EMPTY>	Provides the full path to a script interface to update clients automatically when a FAILURE_INCOMPATIBLE message is generated. The example script upgrade.sh is available as a starting point and is located in the examples directory.
TIBS_UPGRADE_NOTIFY	0	Set to 1 to enable email notifications on automatic upgrade attempts, 0 to disable.
TIBS_MAX_CACHE_SIZE	20	Size of new cache files in GB. All programs that write data to the backup cache use this value as an indicator for when a new cache file should be opened. The limit is enforced after each file is copied to the cache.
TIBS_MAX_CACHE_ALLOCATION_SIZE	20	Size in megabytes of memory buffer used for writing file stream data to the backup cache.
FLDB_SNAPSHOT_ENABLE	1	Set to 1 to enable or 0 to disable. When enabled, the backup server will retain complete file information for every backup in the file lookup database. When disabled, the default information which includes files and necessary directory information will be retained for only the files on the current backup volume. Enabling snapshots provides for a much more robust restore interface, but does require a SIGNIFICANT INCREASE IN STORAGE SPACE for the tape database (default location /usr/tibs/state/tapes). This feature should only be enabled with caution. Contact Teradactyl for additional information on space requirements if you wish to use this feature.
SKIP_CHECKSUM_ERRORS	0	Set to 1 to enable this feature or 0 to disable. Experimental flag which allows checksum errors to be ignored on files that are being removed from the current backup volumes.
DEFAULT_ENCRYPTION	0	Set to 1 to enable encryption of network data transfers by default. Set to 0 to disable encryption of network data transfers. Clients may override the default encryption with the ENCRYPT_ENABLED option in the clients **tibs.ini** file.
KEY_STRENGTH	128	The encryption key length to use. Possible values are 128, 192, and 256.

Settings for tibs.conf use with AFS®

Parameter	Defaults	Description
DEFAULT_AFS_CELL	mycell.com	Default AFS cell to use in place of -C cell options.
AFS_AUTO_GEN	0	Set to 1 enables the update of **.backup** volumes just before incremental backups are run. The .backup volumes are generated automatically during full network backup processing.
TERA_VOS	NONE	Specific location of the **vos** command.
TERA_KLOG	NONE	Specific location of the **klog** command.
TERA_VOS_LOCALAUTH	0	Set to 1, this option adds the -localauth option to AFS commands. The backup server must also be configured as an AFS fileserver, to allow TiBS programs to run without an admin token. This option, allows AFS backups to be completely automated.
TERA_VOS_ENCRYPT	0	Set to 1 to enable the use of the -encrypt flag to AFS vos commands called by TiBS.
TIBS_MARK_AFS_INCR	1	Automatically tags unchanged AFS volumes so that subsequent backups and queries do not need to check them.
ENABLE_AFS_HASHING	0	For sites that maintain tens of thousands of AFS volumes, this parameter enables hashing of volumes names into manageable subdirectories. WARNING: This parameter should not be changed without first re-organizing any existing cache data. Contact Teradactyl for assistance.
AFS_RESTORE_CREATION	0	Specifies how to restore the Creation field on a volume. Possible values are: 0: ignore, older versions of vos do not support this feature 1: dump, preserver the creation date in the dump file 2: keep, do not modifiy the creation date 3: new, assign the create date?
AFS_RESTORE_LASTUPDATE	0	Specifies how to restore the Last Update field on a volume. Possible values are: 0: ignore, older versions of vos do not support this feature 1: dump, preserver the creation date in the dump file 2: keep, do not modifiy the creation date 3 new, assign the create date
AFS_BACKUP_READONLY	0	1 to enable, 0 to disable. If enabled, a single copy of each AFS readonly volume will be maintained by the backup server.
AFS_READONLY_CACHE	<EMPTY>	The backup of AFS readonly volumes is currently

		supported for a single AFS cell to a single backup cache location. If you enable the backup of readonly volumes, you must specify the cache location here.
AFS_READONLY_CLASS:	<EMPTY>	The backup of AFS readonly volumes is currently supported for a single backup class. If you enable the backup of readonly volumes, you must specify the backup class here.
AFS_READONLY_GROUP	<EMPTY>	The backup of AFS readonly volumes is currently supported for a single backup group. If you enable the backup of readonly volumes, you must specify the backup group here.

UNIX® Program Pathnames Defined in tibs.conf

All TiBS scripts use absolute pathnames to run common UNIX® programs. This is to control execution in the event that the **$PATH** environment variable is changed. Some sites have more than one version of these programs (e.g. the OS vendor version and a gnu version). The installation program will attempt to find a version of each program at install time.

UNIX® COMMAND	TYPICAL VALUE
TERA_AWK	/bin/awk
TERA_CAT	/bin/cat
TERA_CP	/bin/cp
TERA_DATE	/bin/date
TERA_EGREP	/bin/egrep
TERA_FGREP	/bin/fgrep
TERA_FIND	/usr/bin/find
TERA_GREP	/bin/grep
TERA_HOST	/bin/host
TERA_LS	/bin/ls
TERA_KILL	/bin/kill
TERA_MAIL	/usr/bin/mailx
TERA_MKDIR	/bin/mkdir
TERA_MV	/bin/mv
TERA_NOHUP	/usr/bin/nohup
TERA_PS	/bin/ps
TERA_RM	/bin/rm
TERA_SED	/bin/sed
TERA_SLEEP	/bin/sleep
TERA_SORT	/bin/sort
TERA_TAIL	/usr/bin/tail
TERA_WC	/bin/wc

Defining the Backup Cache

The file **`state/caches.txt`** contains a list of all hard drive partitions that make up the backup cache. It is best to create only one file system on each hard disk that is used. You may need to create more than one file system per hard drive depending on the size of the hard drive and the limitations of the backup server's operating system. The format is:

```
cachepath|max_balance|max_clear
```

`cachepath:` The absolute path of this component of the backup cache.

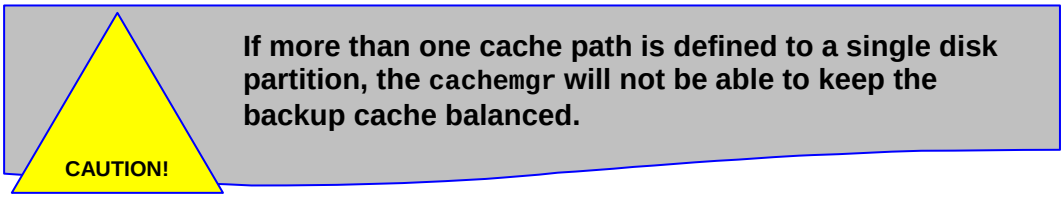
`max_balance:` The percentage (0-100) of used space that the `cachemgr` will allow on this partition before trying to balance the cache. If automatic balancing is disabled, an alert will be generated.

`max_clear:` The percentage (0-100) of used space that the `cachemgr` will allow on this partition before trying to clear volumes. This number should be greater than or equal to `max_balance`. If automatic clearing is not enabled, an alert will be generated.

Example for caches.txt:

```
/cache0|95|98
/cache1|95|98
```

Automatic balancing will occur, if enabled, until no volume can be found to balance, or all the cache partitions have gone over the `max_balance` limit. If cache clearing has been disabled, an alert will be sent to all defined administrators. If you are not using the clearing feature, set the value of `max_clear` equal to `max_balance`.


CAUTION!
If more than one cache path is defined to a single disk partition, the cachemgr will not be able to keep the backup cache balanced.

Introduction to the Disk Library Interface

TiBS now supports an optional Disk Library Interface (DLI). The DLI extends the capabilities of the backup cache by allowing any backup volumes to be stored on disk in addition to, or instead of, being stored on tape. The DLI is an extension of the backup cache. Storage of all disk library backups is maintained in a sub-directory within the cache partition where they are generated. Significant additional disk storage is required to maintain backup data within a disk library. A simple backup cache typically requires about 10-20% of the size of the primary storage area size to function properly. A typical disk library will require a minimum of 50% of the primary storage area size. The amount of disk required to maintain a disk library can be many factors higher depending on data retention requirements.

Overview of the Cache Layout

Within each cache partition are names of the clients that are currently located in that partition of the backup cache. All of the cache backup volumes for each client reside within a client's cache directory. The following files are located in each sub-directory for each client volume:

`full.tdb:`	TiBS backup volume database. Stores meta data for files and directories since the last backup.
`full*.dat:`	TiBS backup volume file data. These files contain the actual file data along with checksum information used to track the movement of file data through backup server processing.
`full.fldb:`	The file lookup database for the current cache backup volume. This file contains summary information about the actual file data in the cache backup volume. Each time a cache backup volume is copied to tape the file lookup database is copied to the tape database.
`full.lst:`	A `vnode` listing for an AFS® backup volume used as a placeholder for backups.
`lock0:`	Backup/Restore lock for the volume.
`lock1:`	The tape writer lock for the volume.
`pending:`	Flag file indicating a volume should be written to tape.
`pending.entries:`	Listing of currently written tape volumes for multiple tape volume backups.
`temp*.*:`	Temporary files used when processing a backup job.

Enabling Fast Cache Recovery

If a backup cache hard drive fails, the cache recovery process normally can be performed by rebuilding the cache from backup tapes. This can take a long time. Fast cache recovery allows the cache to be restored to a previously known state by maintaining copies of each cache volume after any lower level synthetic backup completes successfully and when the volumes are small and contain no file data. By retaining this data in another location on the server, a failed disk or the entire cache state can be reset to the last known lower level backup state. The cache can then be rebuilt by executing a new incremental backup on each backup client in the failed portion of the cache. This may generate a significant amount of network traffic, depending on the type of schedule that is implemented.

Fast cache recovery does require extra server hard disk space since it maintains one or more copies of each clients backup database file (`full.tdb`). Plan on ½ of 1% of the total backup size for each lower level backup you plan to support. For example, a 3 level backup (Monthly, Weekly, Daily) would require two copies of each database to be maintained in the recovery section, or 1%. Plan to add a partition to the backup server that is 10 gigabytes for each 1 terabyte of data that you intend to backup in this example.

To enable fast cache recovery update the `CACHE_FAST_RECOVERY_DIR` option in `tibs.conf` with the directory or mount point for the location where you plan to store this data:

```
CACHE_FAST_RECOVERY=/usr/tibs/recovery
```

Storage of Disk Library Backups

The Disk Library Interface (DLI) stores backup data in a top level sub-directory named `tibsdl` within each backup cache. The directory structure within the disk library is similar in format to the tape database, except that at the lowest level directories, files are stored as they are in the backup cache. This includes the meta data file `full.tdb`, and any supporting data stream files, `full*.dat`.

Defining Tape Drives

If you intend to store backup data on tape, you will need to configure the tape devices and medium changers that are available to TiBS. The tape drive configuration file **`state/drives.txt`** contains state information about all the tape devices available to the server. Edit this file by hand to add or remove tape drives. The format is:

```
device|pool|tapeno|offset|status|device_group|pid|atlidev|atli_id|access|blocksize
```

`device:`	Device pathname used for this tape drive (e.g. `/dev/rmt/0n`). Use the no-rewind device for your operating system. If you want to compress data to tape, use the compression device that is also no-rewind.
`pool:`	Media pool that a mounted tape belongs in. (**`unused`** if off-line)
`tapeno:`	Tape number of the currently mounted tape. (0 if off-line)
`offset:`	The current offset of the mounted tape. (0 if off-line)
`status:`	The current status of the tape drive. (0=off-line, 1=available, 2=busy, 3=reading, 4=writing, 5=rewind, 6=file skip forward, 7=file skip backward, 8=failed)
`device_group:`	The device group (1-9) that this tape device belongs in.
`pid:`	The process ID of the current tape daemon. (0 if none)
`atlidev:`	The device path for the tape library robotic arm that is used to service this tape device. (0 if the tape device is not part of a tape library). The device name is a TiBS specific symbolic link (see Configuring Tape Library Devices below).
`atli_id:`	The tape library id for this tape device (Found with `tibsmnt -q`). For manual mode mounting, use numbers beginning with one to signify each unique tape technology.
`access:`	Current access of the tape drive (0=read-only or off-line, 1=write-only).
`blocksize:`	The block size in kilobytes of the currently mounted tape or 0 if no tape is mounted.

To add a new tape drive in device group 1 with no tape library automation, use the following format:

```
device|unused|0|0|0|1|0|0|1|0|0
```

An example configuration of a tape device in a tape library:

```
device|unused|0|0|0|1|0|/dev/atli0|256|0|0
```

Use of Tape Device Groups

Tape drives that are on the same device chain (e.g. the same SCSI channel) belong to the same **`device_group`**. This will prevent multiple (inefficient) write requests to the same device chain. Use more than one device group if the backup server has tape drives on more than one device chain, or a single device chain that can handle more than one tape drive efficiently. All tape devices must belong to a non-zero device group.

Configuring Tape Library Devices

TiBS uses the convention `/dev/atliX` (`/vdev/atliX` for Panther) to specify the device path for tape library devices. This will typically be a symbolic link to the actual device path that must be created manually. For example, on a Sparc system running Solaris:

```
# ln –s /devices/pci@1f,4000/scsi@3,1/jb@2,0:jb /dev/atli0
```

On Linux:

```
# ln –s /dev/sg0 /dev/atli0
```

The device path `/dev/atli0` can then be used in the `atlidev` field for the definition of tape devices here, and later when defining media pools that use tape.

Configuring Virtual Devices for MAC OSX

MAC OSX does not support traditional UNIX® special devices for tape and medium changes devices. TiBS uses a special directory `/vdev` located on the root file system of the backup server. Within this directory, each file contains the SCSI ID of a tape or medium change device. You must use `tapeX` and `atliX` as the names of the devices, with the first device at X=0, for example use the following for two tape devices and a single robotic arm:

```
/vdev/tape0
/vdev/tape1
/vdev/atli0
```

If the first tape device is located at SCSI ID=1, then `/vdev/tape0` will be a regular file with a single line containing a "1". Note that the 1 must be followed by a new line character. You can then use these device names when configuring TiBS `drives.txt` and `labels.txt` configuration files.

Tape Block Size Considerations

The size of a tape block is defined as `DEFAULT_TAPE_BLOCK_SIZE` in `tibs.conf` in kilobytes. The default value, if not specified, is 256 kilobytes. Some newer tape devices may operate more efficiently with a block size of 512 kilobytes or more. TiBS currently supports block sizes up to 1 MB. For larger tape block sizes, the default memory allocation limits of the operating system may need to be modified. Contact Teradactyl if you wish to use block sizes greater than 1 MB.

Overview of Backup Classes, Groups and Media Pools

TiBS uses **backup classes** to define **backup groups** and **media pools**. A *backup class* consists of one or more backup groups and the associated media pools used to store backup volumes. Many sites will only need one backup class. In general, if all of the data that is being backed up is on the same schedule and can be stored on the same media, a single backup class will be needed. Generally, additional classes are needed if a subset of the data needs to be stored separately, if it is to be backed up on a different schedule, or if there are varying data retention policies. A typical example name for a backup class might be `mycompany` or `mydivision`. *Backup groups* are used to specify sets of common backup clients. The most common use for backup groups is by operating system (e.g. `solaris` and `linux`). Within a backup group, TiBS uses default definitions to make configuring new hosts easier. *Media pools* are used within each backup class to define the level backup strategy for the class. They define how backup volumes are stored (on disk and/or tape), the frequency of backup for each level, and retention polices for data. Additional configuration for tape storage includes tape mirroring and offsite management. For example, a three level backup strategy may use media pools named `Monthly`, `Weekly`, and `Daily`. The Daily pools may be retained on disk and tape, while the Weekly and Monthly backups are stored only on tape. The Monthly media pool may be mirrored to tape with one copy of each tape sent offsite once the tape has been filled (for more detailed examples of how to setup media pools see Defining Media Pools). Once all the backup classes, backup groups, and media pools have been defined, administrators can begin customizing each backup group. Administrators can change or add default definitions and can also add hosts and partitions to the backup groups.

Defining Backup Classes

The **`state/classes.txt`** file contains a comprehensive listing of all the classes defined for the backup server. The format is:

```
classname1
classname2
```

Example for classes.txt:

```
# the backup server recovery class
recovery
# all user defined volumes
users
```

All class information is stored in the **`state/classes`** directory of the TiBS server. Classes are used to define client volumes into manageable sets and to define separate **media pools** for the storage of backup data. Media pools are used to store data to disk, to tape, or to both. Data for client volumes that are defined in a class is stored within media pools assigned specifically for that class. Each class that is defined requires an additional subdirectory to store the class specific information (e.g. `state/classes/myclass` for the `myclass` class). Within each class sub-directory, the following files must also be created:

`labels.txt:`	Used to define the valid media pools for the class.
`groups.txt:`	Used to define the valid groups for the class.
`groupname(s).txt:`	A group file for each group defines client volumes for the group.

Defining Media Pools

Media pools are defined for each backup class created. They control how backups are stored on disk and/or tape. Create a **`labels.txt`** file in each `state/classes/classname` directory *for each class defined*. It defines all of the valid media pools that are available for that class. When storing data on tape, a media pool will consist of tapes with the same pool name but different tape numbers. Mirrored tapes are exact duplicates and will use the same tape number. When storing data only on disk, a media pool uses virtual tape numbers. Each virtual tape will contain up to 10,000 volumes before a new tape number is used. When storing data on disk and tape, each tape number and offset used in the disk library is matched to the corresponding physical tape number and offset. In this case, the actual number of offsets available is virtually unlimited. The format for the file is:

```
pool|tag|tape:disk|atlidev|type|frequency|retention|readers|writers|offsite
```

`pool:` Name of the media pool (e.g. `Full, Week,` or `Incr`).

`tag:` Optional media pool to "`tag`". Tags are used when creating multiple level backup strategies. They define the parent/child relationship of media pools (see examples below).

`tape:disk:` Specifies all destination locations for data backed up to this media pool. The tape field can be set to "`mirror`" to enable mirrored tape writes to the pool. If tape mirroring will not be used set this field to "`none`". To disable writing of data to tape set to "`unused`". The disk field should be set to "`disk`" to enable the Disk Library Interface for this media pool or "`unused`" to disable writing of data to the disk library. The combination of "`unused:unused`" is invalid.

`atlidev:` The tape library device (e.g. `/dev/atli0` or `/vdev/atli0` for MAC OSX) used to access tapes for this media pool. At this time, tape storage for a single media pool is limited to a single tape library. If you are using multiple tape libraries on a backup server, you can place more than one media pool into each library. The integer values starting with 1 should be used if there is no tape library available, or if data is stored on disk only. Each integer value is used to represent a different tape technology for the manual loader.

`type:` The specific type of backup this media pool implements.

`frequency:` The frequency in days that the media pool should be updated. NOTE: for best performance it is recommended that the frequency of related media pools be multiples of each other. For example, if a week is seven days then a month is best defined as 28 days and a year as 364 days. This allows TiBS to schedule backups efficiently.

`retention:` Specifies how long in days to keep backup volumes before they are recycled. Once a backup is recycled, its data will no longer be accessible. Use the value 0 to specify backups that should be retained permanently. When a media pool is defined to both disk and tape, data on tape must be retained at least as long as data on disk. Use the retention value to specify the tape retention policy. Then data that is removed from tape will also be automatically removed from disk. The `hostexpire` command has options that can be used to implement shorter retention policies for disk, when using both disk and tape.

`readers:` When a media pool is defined to use tape, this is the maximum number of tapes that can be mounted for read at one time. A value of 0 indicates that any number of tapes may be read, up to the number of available tape drives.

`writers:` When a media pool is defined to use tape, this is the maximum number of tapes that can be mounted for write at one time. A value of 0 indicates that no data may be written to the media pool. This includes both disk and tape.

`offsite:` When a media pool is defined to use tape, this specifies how tapes are moved to offsite. If a media pool uses tape mirroring this field is defined by to '`:`' separated arguments, one

for each tape in a pair of tapes.

The **type** field determines which kind of backup to run:

tincr:	This pool contains true incremental network backup volumes.
incr:	This pool contains cumulative incremental network backup volumes.
partcum:	This pool contains partial cumulative incremental synthetic backup volumes.
flush:	This pool contains cumulative incremental synthetic backup volumes.
full:	This pool contains full or complete backup volumes.

The **offsite** field determines how to manage tapes as they are filled or merged. This field is only valid when a media pool is defined to store data on tape.

active:	A tape in this media pool will remain onsite at all times. Use this value for all media pools for which no offsite policy is used.
archive_full:	A tape in this media pool will be archived and remain on site when it is filled.
archive_merge:	A tape in this media pool will be archived and remain on site when all merges requiring the tape have completed.
offsite_full:	A tape in this media pool will be taken offsite when it is filled. This sends tapes offsite more quickly.
offsite_merge:	A tape in this media pool will be taken offsite when it is no longer required for backup processing. This sends tapes offsite more slowly.

Media Definitions for Roaming Clients

Since roaming clients have no way of knowing what tapes are currently mounted, special files can be defined in each class directory to indicate the current media pool that a roaming client should backup to. These files can also be used by automation programs to determine what media pool to use each day.

ThisFull.txt:	Current full media pool for a class.
ThisDaily.txt:	Current incremental media pool for a class.

Example Media Pool Definitions

Example 1: A Simple Two Level Tape Backup Strategy

This example shows how data can be stored on tape using a simple two level backup strategy. This is a common configuration for the Small Business Edition. New Network Full backups are generated every two weeks and retained permanently. As Full tapes are filled they are scheduled for offsite storage. Daily cumulative incremental backups are stored to a separate `Incr` media pool and retained for 28 days.

Example 1 labels.txt: Two level tape backups

```
# full backups on tape
Full|Incr|none:unused|1|full|14|0|1|1|offsite_full
# incremental backups on tape
Incr|none|none:used|1|incr|1|28|1|1|active.
```

Example 2: A Simple Two Level Disk Backup Strategy

This example shows how data can be stored on disk using a simple two level backup strategy. This configuration is only available with the Full Version of TIBS. After an initial Network Full backup, New Synthetic Full backups are generated every two weeks and retained for two months Daily True incremental backups are stored to a separate `Incr` media pool and retained on disk for 28 days.

Example 2 labels.txt: Two level disk backups

```
# full backups on disk
Full|TIncr|unused:disk|1|full|14|60|1|1|active
# incremental backups on disk
TIncr|none|unused:disk|1|tincr|1|28|1|1|active
```

Example 3: Two Level Tape Backups with Tape Library and Tape Mirroring for Offsite

This example shows how data can be stored on tape using a two level tape backup strategy with mirroring of Full tapes for offsite. All data is stored in a single tape library. The tape library allows for the automatic processing of Full Synthetic backups without the need for operator intervention. This is a common configuration for use with TiBS Lite. After an initial Network Full backup, new Synthetic Full backups are generated every four weeks and retained for up to one year. As Full tapes are filled the mirror copy is scheduled for offsite storage. When all merge processing of a given Full tape has completed, the tape is scheduled for removal from the tape library to make room for more tapes. Daily cumulative incremental backups are stored to a separate `Incr` media pool and retained for two months within the tape library.

Example 3 labels.txt: tape backup with tape library and mirroring for offsite

```
# full backups mirrored on tape, mirror offsite on filled
Full|Incr|mirror:unused|/dev/atli0|full|28|364|1|1|offsite_full:archi
ve_merge
# incremental backups on tape
Incr|none|none:used|/dev/atli0|incr|1|28|1|1|active.
```

Using Tags to Implement Multi-level Backup Strategies

The tag field in a media pool definition is used to mark backup volumes in one pool whenever the backup of another pool completes. A tag is a tape database entry that marks a client volume as completed for a media pool, even though no actual data is associated with the backup volume. For example, when the full backup of a client volume completes, the administrator will want to tag an appropriate midlevel pools to prevent the premature processing of a newly backed up client volume. Tags are normally only required when using more than two backup levels. The exception for two levels is when Full Synthetic backups are used in conjunction with True incremental backups using the Full Version of TiBS and a disk library (see Example 5 below). Tags may be recursive.

Example 4: Advanced Four Level Backups using Tags, Offsite Requirements, and Mixed Media Pools

In the following example of a four level backup, whenever a backup completes to the `Full` pool, both the `Month` and `Week` pools are tagged. Whenever a backup completes to the `Month` pool, the `Week` pool will be tagged. Tags are available for sites that are using advanced scheduling features in TiBS to simulate multiple level backups. If you are not planning on using these advanced features, leave the tag field set to `none` in all media pool definitions.

Both `Full` and `Month` media pools are defined for tape mirroring and offsite storage. To conserve disk space in the disk library, neither pool is retained on disk. Full tapes are retained permanently, while Month tapes are recycled after one year.

The `Week` and `TIncr` media pools are defined for both disk and tape. The data is retained on both disk and tape for 56 and 28 days respectively. All tapes for both of these pools are maintained in a second tape library.

There is a subtle, but important point regarding the use of True incremental backups and Tags. The Tag is necessary to direct the merge processing to find this data when generating Week or lower level backups. However, when a lower level back completes, it will not create a tag entry within the `TIncr` media pool. Doing so may prevent the normal daily network backup to this pool for volumes that have been consolidated that day.

Tags are not used to restore data. They may be expired just like normal tape backup volumes with the `hostexpire` command (see Scheduling Backups). For a more comprehensive listing of example media pool definitions, see the `labels.txt` template file in the `state/classes/test_class` directory where TiBS is installed.

Example 4 labels.txt: Advanced 4 level backup with tags and mixed media

```
Full|Month|mirror:unused|/dev/atli0|full|84|0|1|1|offsite_full:archive_m
erge
Month|Week|mirror:unused|/dev/atli0|flush|28|364|1|1|offsite_full:archiv
e_merge
Week|TIncr|none:disk|/dev/atli1|flush|7|56|1|1|active
TIncr|none|none:disk|/dev/atli1|tincr|1|28|1|1|active
```

Example 5: Special Case Two Level Backup with Tags

This final example illustrates the special circumstance in which a two level backup will require the use of Tags with the Full Version of TiBS and the Disk Library Interface. The `Full` media pool is defined to tape, mirrored for offsite disaster recovery, and recycled after 1 year. The `TIncr` media pool is defined to both disk and tape and retained for 35 days. New Synthetic Full backups are generated every 4 weeks by consolidating `TIncr` backups over a 28 day period into the backup cache, using data stored in the disk library, and then merging data with the previous Full backup on tape. In this special case two level backup, the `Full` media pool uses the `TIncr` pool as a Tag to locate the `TIncr` backup volumes within the disk library for consolidation. The `Full` pool will not create tag entries for the `TIncr` pool upon completion of a backup consolidation.

Example 5 labels.txt: tape backup with mirroring for offsite

```
# full backups mirrored on tape, mirror offsite on filled
Full|Tincr|mirror:unused|/dev/atli0|full|28|364|1|1|offsite_full:arch
ive_merge
# incremental backups on tape
TIncr|none|none:disk|/dev/atli0|tincr|1|35|1|1|active.
```

Defining Backup Groups

Create a **`groups.txt`** file in each `state/classes/classname` directory *for each class defined*. It defines all of the backup groups for the class. The main purpose of backup groups is to allow administrators to specify a large number of client volumes into common sets. One way to use backup groups is to create one for each operating system that is being backed up. The format for this file is:

```
group|port|type
```

`group:`	The name of the backup group being defined.
`port:`	The TCP/IP port used to contact backup clients. If you are using the default port, the value is 1967.
`type:`	The type of backup group being defined.

The valid **`types`** are currently:

`normal:`	Backup groups that process all backups, including full backups, through the disk cache. Backups can be initiated by the backup server or the backup client.
`roaming:`	Clients that initiate backups on their own. Windows® 95/98 clients must be configured as roaming clients. DHCP clients must also be configured in roaming groups.
`afs:`	This signifies Andrew File System backup groups. The port value has no meaning in this case and is set to 0.

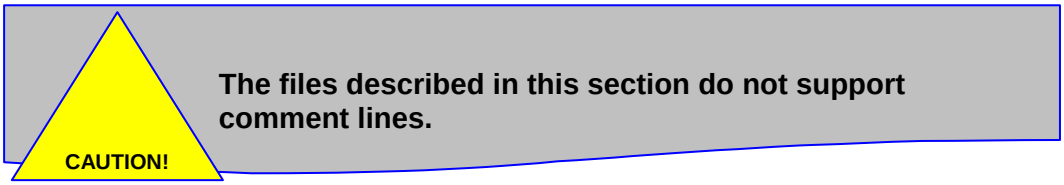
Example of groups.txt:

```
# example groups.txt file
# windows NT clients
winnt|1967|normal
# windows 95/98 roaming clients
win95|1967|roaming
# Andrew File System user volumes
afsusers|0|afs
# some clients that use alternate TCP/IP port
altport|1971|normal
```

Backup Configuration

Managing Client Backup Volumes

The previous sections described all of the server configuration files managed using a text editor. Typically, these files do not change much once the server is setup. Adding and removing client volume definitions is an ongoing process. We recommend using the command line interface provided for making changes to backup group definition files. The command line interface supports updates while the backup server is running and works with the auditing utilities that help backup administrators clearly document their backup plan.



Backup Group Definition File Format

For each backup group defined in each class, a **`groupname.txt`** (e.g. `solaris.txt`) file exists. It defines all of the client volumes in the backup group. The format is:

```
client|volume|location|rulefile
```

`client:` The fully qualified hostname of the backup client.

`volume:` Absolute pathname of the partition or sub-directory to backup.

`location:` The cache location for backup volumes (as defined in `caches.txt`).

`rulefile:` Rule file for this client volume. The default rule is *none*. (See Also Use of Rule Files).

Example client volume entries:

```
ptero.teradactyl.com|/var/spool/mail|/cache0|none
ptero.teradactyl.com|/usr|/cache0|usr
```

Adding New Client Volumes

Define client volumes from the command line using the **`hostadd`** command:

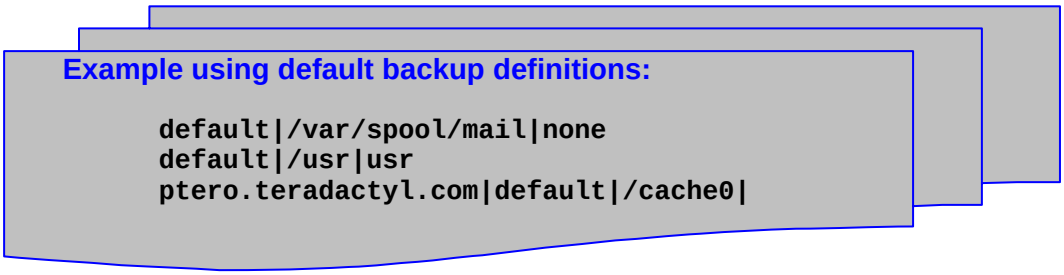
```
# hostadd -c class -g group -n host -p path -r rule -l location
```

All of the arguments except **`-r rule`** and **`-l location`** are required for the command to succeed. If no rule file is passed, the special rule, *none*, is used in the definition. If no cache location is specified the location in the cache is determined as follows:

1. The cache partition with the largest free space is used when adding a host for the first time.
2. If the host already has volumes defined, the current location is used.

Default Backup Volumes

The reserved hostname, **`default`**, defines client volumes for every client defined with the special volume name, **`default`**, within a backup group. Default volume definitions appear at the beginning of a group's definition file. The above example using default definitions would look like:



Default definitions help reduce the complexity of group files for groups of clients that have many similar volumes to backup or omit. It is possible to define clients that do not use the default definitions and keep them in the same backup group. Default backup volume definitions do not contain a cache location field.

Adding New Default Volumes

Define default volumes from the command line using the `hostadd` command with the special **`-n default`** option:

```
# hostadd -c class -g group -n default -p path -r rulefile
```

All of the arguments except the *`-r rulefile`* are required for the command to succeed. If no rule file is passed, the default rule, *`none`*, is used in the definition. Individual clients can then be defined to use the default definition(s) with the **`-p default`** option:

```
# hostadd -c class -g group -n client -p default
```

TIP: Rule files are not supported in default host definitions. The special rules `skip` and `omit` will work and are used for auditing and backup management.

Overriding Default Volume Definitions

Some partitions may require special treatment, such as a custom rule file. If the host has already been defined with default partitions, and the partition definition that you wish to customize is part of the default partitions, you can override the definition for the individual host with the **`-o`** option to `hostadd`:

```
# hostadd -c class -g group -n client -p /usr -r special -o
```

Managing Roaming Clients

TiBS supports the backup of DHCP hosts through the roaming backup group type. These are hosts which do not have static IP addresses. This special group type indicates to the backup server that the IP address for the host cannot be located, and that the roaming client will initiate its own backup, either manually or automatically. TiBS must be installed on roaming clients using install media or by downloading the appropriate install files from the Customer Center at the Teradactyl® website. The TiBS client installation requires a fully qualified client hostname to authenticate the client with the backup server. You can use any hostname (e.g. `roaming1.domain`), but it must be unique to the backup server since only one client can use the authentication files for a given hostname. To add client partition definitions for unqualified hostnames use the **`-F`** option to `hostadd`:

```
# hostadd -F -n roaming1.domain -c class -g roaming -p partition
```

Server Disaster Recovery Configuration

The TiBS install directory contains all the information required to recover the server. For example, if TiBS is installed in the default directory, `/usr/tibs`:

```
# hostadd –c class –g group –n backup_server –p /usr/tibs –r recovery
```

The omit rule file `recovery.txt` is created as part of the backup server installation in `/usr/tibs/state/rules`. You should review this rule file to make sure that it is correct for your installation.

Windows® Registry Backup

To enable this feature, use the `/c/repair` special path procedure for Windows® with the `hostadd` command:

```
# hostadd –c class –g group –n client –p /c/repair
```

This feature allows clients to extract registry information and can detect some corrupted registry files if it fails. The backed up registry information obtained using this process is not currently supported for recovery. Teradactyl strongly recommends that sites that wish to protect Windows® 2000/XP system drives purchase an Open Transaction Manager™ (OTM) license for TiBS. This option will provide for the complete and stable backup of Windows® system drives. With OTM, a failed system drive can be completely restored without the need to perform additional steps to restore the registry.

Removing Client Definitions

To remove all volume definitions for a client completely from the backup server use the **`hostdel`** command:

```
# hostdel –n client
```

Confirmation is required to remove data from the backup cache and to update the network audit database. Removal of an entire client will update the network audit database to reflect that the client no longer requires backups.

To remove a specific client backup volume, use the **`–p path`** option:

```
# hostdel –n client –p path
```

Authenticating New Backup Clients

If you are installing TiBS client software using a local software distribution mechanism, you must authenticate each backup client with a backup server. This can be done from the server with the `hostaudit` utility:

```
# hostaudit –n client –a
```

or from the backup client with the `tera` (`tera.exe` for Windows® clients) program:

```
# tera -a
```

Andrew File System Configuration

The Andrew File System (AFS®) configuration files are located in **`state/afs`**. The primary configuration file, **`cells.txt`**, contains a list of the AFS cells that are being managed by the backup server. Each cell's backup configuration is maintained in a subdirectory, **cell.domain**. The **`afsgen`** script generates the current volume lists for every cell. This script separates all of the volumes it finds into backup groups. In general, you only need to use multiple backup groups if you are managing AFS® volumes in separate classes with different backup schedules or are managing multiple cells. To configure the AFS® groups, edit the **`afsgroups.txt`** file *for each cell* with the format:

```
group|class|location
```

`group:` The name of the backup group to be generated.

`class:` The name of the backup class that this backup group belongs to.

`location:` The cache location for all backup volumes in the group (as defined in `caches.txt`).

Note: Define each class in `state/classes.txt` and each backup group as type `afs` in the class' `groups.txt` file.

For each group there are two files, **`accept.group`** and **`omit.group`**, that are used as filters for the group. Filtering is based on the name of the read-write volume that is to be backed up. The format for these files is:

```
|afsvolume_preamble.
.afsvolume_extension|
|volumename|
any_volumename_with_this_string
```

The file `accept.group` is used to define valid volumes for the associated backup group. If there is no accept file for a group, then the group accepts all volumes. The file `omit.group` is used to eliminate unwanted volumes. If there is no omit file for a group, then a group keeps all of the volumes it accepts. The `afsgen` script generates **`group.list`** files for each group and the file **`nobackup.undefined`** for unassigned volumes. Copies of the list files are placed in the defined class directories as `groupname.txt` and a report is sent to all defined backup administrators. The report shows inconsistencies between the VLDB and the volumes on each fileserver along with information about volumes not defined to any backup group. To remove volumes that do not require backup from this report, add them to the appropriate `omit.group` file or add a line containing each volume name to **`nobackup.defined`**. The format of an AFS® backup group file is different from other group file definitions:

```
fileserver|cell|volname|location|last_update|rulefile
```

`fileserver:` The name of the fileserver this volume currently resides.

`cell:` The cell to which the file server belongs.

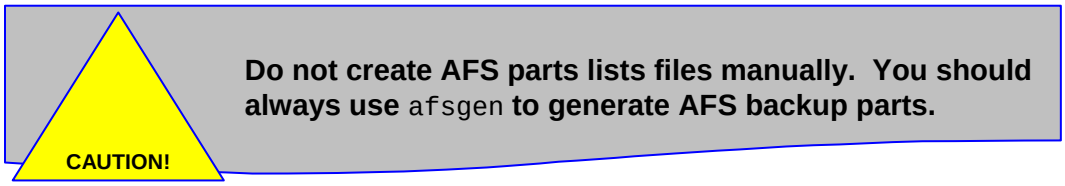
`volname:` The read-write volume name.

`last_update:` The last update time for the volume (`YYYYMMDDhhmmss`).

`location:` The cache location for backup volumes (as defined in `caches.txt`).

`rulefile:` The rule file to use for this volume (currently only *none*).

TIP: It is best to run `afsgen` manually whenever making configuration changes to make sure all AFS® volumes get assigned properly.



Do not create AFS parts lists files manually. You should always use `afsgen` to generate AFS backup parts.

Configuration for AFS .readonly Volumes

TiBS now supports the backup of read-only volumes within a single AFS cell. To enable the backup of these volumes, check the AFS settings in `/etc/tibs.conf` for the backup cache, backup class, and backup group definitions. You must still define the backup class (if you are not using an existing class) and backup group, which must be a separate group defined only for read-only volumes. You can then use the filter files, `accept.readonly` and `omit.readonly` to further specify the volumes that need to be backed up.

Automating AFS® backups

By default, TiBS requires ADMIN rights using `klog admin` in order to run certain `vos` commands. The steps to automating AFS backups include:

1. Update `the state/tibs.conf` file to enable the `-localauth` flag to all AFS commands:

```
TERA_VOS_LOCALAUTH=1
```

2. Securely install your cell's `KeyFile` file:

   For: OpenAFS: `/usr/local/etc/openafs/server/KeyFile`
   For: IBM-AFS: `/usr/afs/etc/KeyFile`

Alternatively, you may configure your cell to allow `root` on the backup server(s) to access the cell as `admin`.

Multiple Cell Support

TiBS can support multiple AFS® cell configurations with the following constraints:

1. Each cell must be configured to a separate class with separate media pools.
2. Only one cell may be automated by making the backup server a "dummy" fileserver in that cell.
3. All other cells require admin authentication before backups are run.

These issues will be addressed in a future release of TiBS to allow full automation of multiple AFS cells to a single set of media pools. Please contact Teradactyl® for details if you intend to support multiple cells to a single server.

AFS® Configuration Example

Assume an AFS cell, `cell.com`, has named all of its system volumes as `system.*` and all of its user volumes as `user.*`. Assume also that backups for system volumes go to different sets of tapes than the user volumes. First, the site manager configures `state/afs/cells.txt` with the name of the site's AFS® cell:

```
cell.com
```

The manager then creates two classes, system and user in the `states/classes.txt` file:

```
system
user
```

The manager then creates two backup groups:

`afssystem`, in the `system` class's `groups.txt` file:

```
afssystem|0|afs
```

`afsuser` in the `user` class's `groups.txt` file:

```
afsuser|0|afs
```

With the classes and groups configured, the manager now defines `state/afs/cell.com/afsgroups.txt`:

```
afssystem|system|/cache0
afsuser|user|/cache1
```

The manager then creates `state/afs/cell.com/accept.afsuser` file for all AFS® volume names beginning with `user`:

```
|user.
```

The manager then creates `state/afs/cell.com/accept.afssystem` file for all AFS® volume names beginning with `system`:

```
|system.
```

Since all volumes are backed up, there is no `omit.user` or `omit.system` file in `state/afs/cell.com` at this time. Now the manager runs `afsgen` to test the configuration. After a few minutes, the program completes. The `state/afs/cell.com/nobackup.undefined` file now contains the following:

```
root.afs
root.cell
```

These are considered system volumes, so the manager updates `state/afs/cell.com/accept.system`:

```
|system.
|root.
```

The administrator then runs `afsgen` again, and the `nobackup.undefined` file is empty. The AFS® configuration for backup is completed and tested.

Use of Rule Files

Each client volume has a **rule file** defined to eliminate unwanted files and directories from backup. Each rule file contains the relative pathnames of files and directories. Data defined in a rule file is not to be copied from the client volume. The default rule file is **none**, signifying a backup of all data on a client volume. The following special rules do not have an associated rule file:

`file:`	Signifies the backup of a single file. The file is backed up every time it changes, and the current version is maintained in the cache.
`none:`	All data in client volume copied to backup volumes. This is the default rule.
`omit:`	Do not backup a client volume (used for auditing).
`persist:`	All data for the client volume is retained in the backup cache. Typically used for small, critical client volumes so that restores can be done without mounting tapes (e.g. Windows® registry backup).
`skip:`	Temporarily skip a client volume (tracked in reporting).

The format is a relative path signifying a file or directory to remove from backups. *Be careful when creating and using rule file definitions.* There is currently only one valid special character '**`*`**' which can only be used by itself such as */tmp/** but not **.tmp*

The special character '*' can be used to omit all of the files from a subdirectory but not the directory itself. The empty directory will be created when performing a restore.

Example rule file entries:

```
*/.netscape/cache/*
user/large_logfile
```

All rule file definitions are located in **`state/rules`** as **`rulefile.txt`**. Each time you create a new set of rules, add a file to this directory with the set of rules you desire. You may then reference the name, **`rulefile`**, as a rule file argument in any client volume definition.

Note: Windows® clients still interpret file and directory names as case insensitive. The Windows® client does not check case when enforcing rules. For example, the rule `winnt` would match any combination of upper and lower case spellings such as:

```
WINNT      winnt     WinNT
```

Auditing Clients

TiBS comes with auditing procedures to check the status of backup clients on the network. These programs can augment existing procedures or become part of a new auditing procedure for a site. They help backup administrators ensure all data for the site is backed up properly. The programs work with class definitions to provide a clearly documented backup strategy, including those clients or volumes that do not require backup. There are two steps to completing a full site audit:

Network Audit:	This audit processes sub-net information and updates the client audit file.
Class Audit:	This audit compares client definitions with the client audit file and the client.

Sub-Directory Considerations for Auditing

Sometimes clients may have partitions that require only partial backup. These definitions are not audited on backup clients but will show up as an audit error (`NO_PARTITION`). There is a mechanism for ignoring these audit errors. There is no current support for auditing subdirectory definitions. The safest way to ensure that all clients that require a subdirectory backup (e.g., `/var/mail`) are defined properly is to use default definitions (see Default Backup Volumes).

Network Auditing

This audit procedure tracks changes in hostname definitions on a network over time. A significant manual effort may be required running this audit the first time. Subsequent audits will reuse information, reducing the time for this procedure. Network auditing works on class C sub-nets. The configuration file for sub-net definitions is **`state/subnets.txt`** and has the format:

```
sub-net|start|end|convert
```

`sub-net:`	The class C sub-net for this range (e.g. 128.2.236).
`start:`	The start address for this range (0-255).
`end:`	The end address for this range (0-255).
`convert:`	0=no conversion, 1=lower case, 2=upper case.

A sub-net can be broken up into several ranges when only portion(s) of the sub-net are audited:

```
128.2.236|0|63|1
128.2.236|128|191|1
```

The current state of the network audit is in the file **`state/clients.txt.`** The format for this file is:

```
hostname|status|comment
```

`hostname:`	The fully qualified hostname for this client (e.g. `host.domain`).
`status:`	The current audit status (`omit`, `backup`).
`comment:`	Optional comment field for site use.

The **`status`** field determines the state of the client from a backup perspective:

`omit:`	The client at this address does not require backups.
`backup:`	The client at this address requires some or all of its data backed up.

Running the Network Audit

To run a network audit use the **`netaudit`** command:

```
# netaudit
```

The script will report any discrepancies it cannot resolve and prompt the backup administrator to make any necessary corrections. Any hosts that were not resolved appear in the file, **`reports/networks/clients.nodef`**.

> **TIP: Backup administrators may wish to review this information with the appropriate personnel to ensure that hosts omitted at this level do not need any type of backup.**

Resolving Network Audit Errors

`UNKNOWN_STATE: host with 0/0 parts/omits`

The host was found on the network and must be added to the network audit database as either `backup` or `omit`. Select one of the following options to add the host to the network audit file:

```
1) add host for backup
2) add omitted network device
3) add omitted unsupported host (e.g. MacOS)
4) add omitted dataless/cycle server host
5) add omitted with optional comment
<RETURN> ignore host and continue
```

`no_address hostname`

A host exists in the network audit database that does not currently have an assigned IP address. This typically happens when a machine is no longer in service. Answer *`yes`* to remove the host from the audit file. This will also remove any host definitions from the backup definition files. Any other response will leave the host status unchanged.

`UNKNOWN_HOST: at X.X.X.X responding to ping`

A response to an undefined host that was found at IP address X.X.X.X. The host could be a DHCP host or an illegally attached or undefined system.

Alternative to Network Auditing

You may not need to run the network audit if your site already has an adequate tracking mechanism for hosts that require backup. Instead, place a list of fully qualified hostnames that require backup in the file, **`state/clients.txt`**, in the form:

```
host.domain|backup|
```

Class Definition Auditing

Class auditing takes information from the network audit file, `state/clients.txt`, and identifies any discrepancies between backup clients and their definitions. Specifically, it looks for:

Hosts that have class definitions but are not defined for backup.

Partitions that have not yet been defined are found on a host that is defined for backup.

Partitions that are not found on the host, but are defined for backup.

The audit displays all of the differences that it finds, but does not take corrective action. Correct all of the audit errors found at this level by running `hostadd` or `hostdel` for the appropriate definitions. Run the **`classaudit`** command to generate the current discrepancies:

```
# classaudit
```

Detailed results of the audit are in the **`state/audit`** subdirectory. This directory is re-created every time a class audit is run.

Resolving Class Audit Errors

`ERROR: INCOMPATABLE version TiBS 2.x.x.x on host.domain`

The current client software is out of revision with the backup server. Install the appropriate revision of TiBS onto the client. Alternatively, you can add the client version, "`TiBS 2.x.x.x`" to the file `state/versions.txt` to allow the server to continue to communicate with the client.

`NO_AUTH full unknown_group host.domain unknown_path`

The client host has not yet been authorized to the current backup server. Use the `hostaudit` command to authorize the client for backup services (see Authenticating Clients).

`OMIT_HOST: hostname has part path defined`

The backup server contains a definition for a host that does not require backup. This message indicates that there are still parts defined for the host. Change the audit state in `state/clients.txt` from *`omit`* to *`backup`* if the data is still required for backup. Otherwise, delete the class definition with `hostdel`.

`WATCHDOG_TIMEOUT full unknown_group host.domain unknown_path`

The host timed out from the audit request. The host may have been rebooted, or taken off of the network during the `hostaudit`. Verify a proper network connection and retry the audit.

`NOT_DEFINED: host host.domain partition path`

The backup server does not have a definition for the given partition. Use the `hostadd` command to define the partition. Use the **`-r omit`** option if the partition does not require backup.

NO_PARTITION: host host.domain partition path

The host did not detect a partition defined for backup. This error occurs on all non-partition (sub-directory) backups as well. Determine if the path is a partition or sub-directory. Use the `hostdel` command to remove obsolete partitions from backup. If the path is a sub-directory, place a copy of the message in the file, `reports/audit.flt`. The sub-directory will no longer cause an audit error.

NO_REFERENCE: host.domain has auth files with no reference in clients.txt

A host that has not been defined for backup or omit in `state/clients.txt` has authentication files in `/usr/tibs/auth`. The system may have been removed from service. It is possible that this was an authentication request from an invalid or mistyped hostname. You can remove the `host.domain.auth` and `host.domain.dat` files from the `/usr/tibs/auth` directory. Be careful not to delete other client's `.auth` and `.dat` files. Alternatively, this could be a newly authenticated backup client that has not yet been defined to the backup server. In this case, use the `hostaudit` and `hostadd` commands to query the new client and define it to the appropriate backup classes and groups.

WARNING: Auditing host.domain in roaming group

This warning indicates that the host that is being audited is a member of a roaming group. If the host has a static IP address, then the audit may proceed. Otherwise, the audit will fail with `CLIENT_REFUSED.`

WARNING: Cannot determine official hostname of host.domain

The host name is no longer valid. If this is a roaming client, it may be setup through DHCP. The server cannot perform the audit, because it does not know the IP address. You can remove this error message from the audit reports by placing a copy of the message in the `reports/audit.flt` file. If the host has been removed from service, then it may be necessary to remove it from the backup definitions using `hostdel`.

WARNING: No parts found for host host.domain

The host has been defined for backup but there are not actual data partitions defined. Use the `hostadd` utility to add the necessary partitions or the `hostdel` utility if the machine really does not require backup.

Automating Audit Processing

The **`genreport`** command can automatically generate a daily network and class audit report and e-mail the results to backup administrators as a **`cron`** job:

```
30 1 * * * /install_bin/genreport –n –p –m BackupAudit > /dev/null 2>&1
```

Other Uses of the Audit Program

Authenticating Clients

Clients are authenticated at install time. However, if the authentication files are lost or corrupted, the client may need to be re-authenticated. From the backup server, use the **-a** flag to `hostaudit` to re-authenticate a client:

```
# hostaudit -n client -a
```

To re-authenticate clients from the server after a rebuild us the **-r** flag to `hostaudit` to remove server authorization files before authenticating a client:

```
# hostaudit -n client -r
```

Debugging Clients

You can view a backup client's current log file with the **-d** option to `hostaudit`:

```
# hostaudit –n client –d
```

Listing Client Partition Information

You can view a backup client's partition information in a df-style format with the **-q** option to `hostaudit`:

```
# hostaudit –n client –q
```

Viewing/Updating Client Configuration Files

You can view a backup clients' current `tibs.ini` configuration file with the **-c** flag to `hostaudit`:

```
# hostaudit –n client –c
```

You can update client configuration file information with the **-u** and **-w** options to `hostaudit`:

```
# hostaudit –n client –u parameter –w value
```

Parameter	Value/Range	Description
MY_HOST_NAME	host.domain	Used to correct for an invalid or changed hostname on the backup client. Once the hostname is updated, the client authentication files are removed and a re-authorization is performed.
MAX_LOG_SIZE	10 – 2048	Maximum size in kilobytes of client log file. When the size limit is reached the log file is reset. The default is 32 kilobytes.
ENCRYPT_ENABLED	0 or 1	Disable/enable network encryption on this client. This parameter overrides any default encryption settings on the backup server. If the parameter is missing the default encryption mode is taken from the backup server.
IGNORE_SERVER_IP_CHECK	0 or 1	Set to 1 to allow client to accept connections from an alternate server IP address. Useful when a secondary network is being configured for backup traffic.
OTM_CACHE_SIZE	10 – 20480	Size in megabytes of the OTM cache. The space allocated must be available on the client's system drive the first time OTM runs. Once the cache space has been allocated, it remains allocated in the `tibs` client directory (Windows® NT/2000/XP/2003 clients only).
OTM_WAIT_TIME	1 – 30	Wait time in seconds for OTM startup. OTM waits this many seconds for a period of inactivity before enabling open file management (Windows® NT/2000/XP/2003 clients only).
OTM_ENABLED	0 or 1	Disable/enable OTM during backup processing. If OTM is disabled, then open files will not be backed up (Windows® NT/2000/XP/2003 clients only).
OTM_THROTTLE_DISABLE	0 or 1	Set to 1 to allow OTM to process backups faster by not throttling the backup traffic when users or other applications are busy accessing the system.
OTM_ALLOW_ACTIVE	0 or 1	Allow backup to run if OTM does not start up. If OTM does not start up, then open files will not be backed up.
OTM_CACHE_PATH	Pathname	Used to redirect the OTM cache to an alternate location (Windows only). A UNIX style pathname (e.g. `/d/tibs`) is used to specify the directory where the OTM cache should be located. The client converts this value to a Windows style pathname (e.g. `D:\tibs`). This is primarily used when a larger OTM cache is required and there is not enough space on the system drive.
REPORT_LOCAL_FS_ONLY	0 or 1	Set to 1 to allow only local file systems to be reported by hostaudit –q. By default, all file systems including network file systems are reported.
RUN_LOCAL_MODE	0 or 1	Set to 1 to allow roaming clients to use the local mode backup by default. This is equivalent to the –l flag for tera/tera.exe. Used to allow roaming clients to perform backups when the server cannot connect to the client's services. Typically used to work around firewall connection problems.
VERBOSE	0-10	Enables more detailed information to be sent to a client's `teralog.txt` file for debug purposes.

Note: **If the client is not using default port, 1967, and is not yet configured for backup, use the `-P port` option to `hostaudit` to contact the client. Clients that are configured determine alternate ports automatically.**

Viewing Client Revision Information

To see what client operating system and version of TiBS a client is currently running use the **`-s`** option to `hostaudit`:

```
# hostaudit -n client -s
```

Media Management

Backup Cache Management

The `cachemgr` daemon monitors the status of the backup cache and will optionally take corrective actions to prevent the cache from overflowing. It sends alerts to all defined backup administrators when it is unable to resolve a potential overflow, and manual intervention may be required. The best way to avoid a cache overflow is to allow `cachemgr` to take corrective actions automatically.

Automatic Cache Balancing

Set the `CACHEMGR_BALANCE` flag in `tibs.conf` to 1 to enable automatic cache balancing. This allows `cachemgr` to move volumes from one cache partition to another, freeing needed disk space on a filling partition. Balancing will only work if there is more than one cache partition defined. Balancing will stop when all available cache partitions exceed their `max_balance` percentage requirement defined in `caches.txt`.

Automatic Cache Clearing

Set the `CACHEMGR_CLEAR` flag in `tibs.conf` to 1 to enable automatic cache clearing. This allows `cachemgr` to replicate incremental data to a lower level tape volume. The server should be able to access a valid tape at all times (either permanently mounted or through a tape loader). Once incremental data is replicated to a lower level volume the file contents of the cache backup volume are removed, freeing the needed disk space in the cache.

Manual Cache Relocation

To relocate a client's cache backup volumes manually use the **`cachemove`** program:

```
# cachemove –c cachepath –n client
```

All cache moves occur at the client level because all backup cache volumes for a client must be located in the same cache partition.

Configuration with the Disk Library Interface

Cache balancing and cache clearing must be disabled at this time when using the new Disk Library Interface module. The `cachemove` program will be updated to support movement of cache volumes, including the appropriate disk library volumes in a future release of TiBS.

Tape Management

This section discusses tape management issues as they relate to TiBS. The information in this section will not apply to your site if you are not using tape media for backup. The **`tibsmnt`** program manages all changes in a tape drive's state. Run without any arguments, it displays the current state of all defined tape drives:

```
# tibsmnt

Device        Label       Offset    Pid      ID     Status      Access     BSize
/dev/rmt/4n   Apr.00.9    10        15236    256    busy        write       48 KB
/dev/rmt/5n   Tue1.1      143       18789    257    writing     write      256 KB
/dev/rmt/6n   Apr.00.10   1         0        258    available   read        48 KB
```

Manually Labeling Tapes

Tape labeling is now automated through the `tibsmnt` interface. Tapes must be labeled the first time they are used, or if a new name is being assigned. To manually label tapes, use the **`tapelabel`** command. All of these arguments are required:

```
# tapelabel –l pool –n tapenumber -t device
```

The command fails if the `tape number` for the `pool` already exists, or if the current label on the tape is still valid. To re-label a tape the contents of the tape database must already be removed (see Removing Tapes from the Tape Database).

To create a pair of **mirror tapes**, use the **`–m device`** option when labeling:

```
# tapelabel –l pool –n tape_number -t device –m device2
```

There must be a tape mounted in each of two devices when creating a tape mirror. Once the mirror is created, all tape write requests must succeed to both tapes.

Viewing a Tape Label

You can view the current label on a tape with the **`-q`** option to `tapelabel`:

```
# tapelabel –t device -q
```

Mounting Tapes

A tape may be mounted for use in **read-only**, **read-write**, or **write-only** modes. A tape must have physical write access in order for tape writes to succeed.

To mount a tape for **write-only** access, use the command:

```
# tibsmnt –w pool
```

The program will search for the lowest numbered tape in the pool that has not been filled. If none is available it will compute the next usable tape number and prompt for a blank tape. If the tape is already loaded into an available drive it will mount the tape, otherwise it will ask that the tape be mounted. The program will then start a *tibstaped* process to watch for in-coming write requests. This process keeps other programs from accessing the device while it is mounted.

To mount a tape for **read-only** access, use the command:

```
# tibsmnt –r pool –n tapenumber
```

If the tape is already loaded into an available drive it will mount the tape, otherwise it will ask that the tape be mounted. TiBS programs cannot write to tapes that are mounted read-only.

To **dismount** a tape, use the command:

```
# tibsmnt –u pool –n tapenumber
```

Note: When dismounting mirror tapes that have been mounted for writing, it is not necessary to specify the second tape device. Both tapes will be dismounted at the same time.

Recycling Tapes

A labeled tape may be re-used by clearing the contents of the tape database for the tape with the **`-r`** option to **`tapedel`**:

```
# tapedel –r -l pool –n tapenumber
```

Automation programs use this function to clear the entries from tapes written in a previous backup cycle that are being re-used (to recycle tape volumes on a time basis see Removing Older Media Volumes Permanently).

Removing Tapes from the Tape Database

Administrators may manually remove specific tapes from the tape database. Some reasons for this include removing tapes from service permanently and deleting tapes that have failed.

Delete tapes permanently from the tape database with the `tapedel` command:

```
# tapedel -l pool -n tapenumber
```

Do not mount tapes before deleting them. The program will prompt for confirmation. Answer *yes* to delete the contents of this tape from the tape database.

Tape Library Commands

The automated tape library interface package automates most of the tape operations described above. Additional commands are available to manually manipulate tapes within the tape library with the **`tibsmnt`** command:

The current state information of a library can be viewed and used for manual operations or additional script automation with the **`-q`** option:

```
# tibsmnt -q
```

Tapes can be manually relocated from one physical location to the next with the **`-f atli_id`** and **`-t atli_id`** options.

```
# tibsmnt –f 257 –t 4198
```

Tapes can be unloaded to import/export locations in a library by media pool and tape number with the **`-U pool`** and **`-n tapenumber`** options:

```
# tibsmnt –U Full –n 10
```

Erasing Tapes

To erase the contents of a tape use the **`-e`** option to `tapedel`:

```
# tapedel -e device
```

The program reads the tape label and removes it from the tape database. The backup administrator must answer *yes* to the confirmation request to erase the tape. This command only works on tape devices that are not currently mounted for TiBS operation.

Scanning Tapes

To **quickly scan** the contents of a tape use the **`tapescan`** command:

```
# tapescan –t device
```

This command will only work on tape devices that are not mounted for TiBS access. This command scans the volumes on the tape and compares them with the current tape database. Any differences are reported to the screen.

To perform a **low level scan** use the **`–s`** flag:

```
# tapescan –t device –s
```

This will simulate the restore process on every volume on the tape. This can take several hours.

Backup Operation

There are several methods for running backups. Below is a summary of the commands:

afsback:	Server initiated program for `afs` network backups.
tera:	Client initiated program for `roaming` network backups.
teramerge:	Server initiated program for processing synthetic lower level backups.
tibs:	Server initiated program for `normal` network backups.

While a backup operation is being performed, the cache backup volume is marked as `busy`. Upon completion of each client volume backup, the cache backup volume is marked as `pending`. A *`tibstaped`* process writes the data to tape. The *`tapemgr`* will attempt to mount an appropriate tape for write-only access, or will send an alert to defined backup administrators requesting a manual mount if no valid tape is currently mounted. No other backup or restore operations on a client volume will run while the cache volume is pending or busy.

TIP: The `-q` query option is excellent for pre-determining the actions a set of arguments to a backup program will perform. Use the query flag to build the appropriate command and then remove the query flag to perform the operation.

Server Initiated Backups

Backup `normal` backup groups with the **`tibs`** command. The command has one required argument that specifies the media pool that will contain the backup volumes. The type of backup performed (full, cumulate incremental, or true incremental) depends on how the media pool is defined (see Defining Media Pools).

```
# tibs -l pool
```

This command will process all client volumes defined in any backup groups that are defined as `normal` in all classes for which the pool is valid. If a client volume has already been written to a pool, the volume is not backed up again (to add more than one backup volume for a client volume to a media pool, see Expiring Backup Volumes from Media Pools). Any combination of the following arguments is also acceptable:

```
# tibs -l pool -c class -g group -n client -p path
```

Each additional argument filters what volumes are processed. For example, to backup all defined volumes on a particular client use:

```
# tibs -l pool -n client
```

Full Network Backups

The first full backup in TiBS and TiBS Lite, and all full backups in TiBS Small Business Edition come from the backup client over the network. Data in the backup cache is removed after the full backup is written to tape. To perform a full backup over the network, define the media pool as type `full`.

```
# tibs -l pool <additional options>
```

True incremental Network Backups

True incremental backups gather information from backup clients containing only files that have changed since the last successful backup. The backup volume is stored within a disk library and optionally copied to tape and then the file data is removed from the backup cache. True incremental backups should only be used in conjunction with the Full Version of TiBS and the Disk Library Interface module. To perform a true incremental backup, use a media pool that is defined as `tincr`:

```
# tibs -l pool < additional arguments>
```

Cumulative Incremental Network Backups

Cumulative incremental backups gather information from backup clients containing only files that have changed since the last successful backup. TeraMerge® technology integrates these files into the backup cache. The cache retains only the files needed to restore the client volume to the current state. To perform a cumulative incremental backup, use a media pool that is defined as `incr`:

```
# tibs -l pool < additional arguments>
```

Synthetic Lower Level Backups

Create full, cumulative, or partial cumulative backups with **`teramerge`** without having to contact backup clients. The program generates a current backup for each client volume that is not currently in the media pool by merging the current cache data with any previous backups that may be required to complete the process.

```
# teramerge -l pool <additional options>
```

The new backup volume resides in the backup cache, marked as pending, until written to tape. If necessary, file data is retained in the cache until any lower level merge processes are completed. All backup groups process during a merge.

A merge can **process in parallel** by merging individual tapes with the **`-t tapenumber`** option:

```
# teramerge -l pool -t tapenumber
```

When processing merges in parallel, an additional call to `teramerge` with the **`-F`** flag must be made to finish processing of all volumes:

```
# teramerge -l pool -F
```

Client Initiated Backups

Client initiated backups are performed with the **tera** (**tera.exe** for Windows®) program from the backup client. This program contacts the backup server to obtain configuration and job information and identify the client's IP address to the backup server. The server then attempts to backup the client using the clients installed backup service. To run a client backup:

```
# tera
```

For clients that cannot be contacted directly by the backup server (usually because they are behind a firewall), use the **–l** flag to enable local mode backups:

```
# tera -l
```

This flag should be used only by users with root access (UNIX) or as a defined Administrator (Windows). Otherwise, the client backup may fail for any number of security and access reasons. For clients that are always roaming because they do not have a static IP address, you can enable the local mode backup by adding the entry in the client's tibs.ini configuration file:

```
RUN_LOCAL_MODE=1
```

To specify the backup of a single client volume, use the **-p path** option to `tera`:

```
# tera –p path
```

To see what type(s) of backups the server will process, query the backup server with the **-q** flag:

```
# tera –q
```

To see the state of all currently defined client volumes, including last backup time, use the **-Q** flag:

```
# tera –Q
```

Use of Client Side Pre/Post Processing Scripts

UNIX® and Windows® clients can run scripts before and after a backup operation. The location for the scripts is the TiBS client install directory. The following scripts are available:

UNIX®	Windows®	Function
`preproc`	`preproc.bat`	Performs functions such as creating a snapshot or shutdown of an application before a backup is performed. If the pre-processing script fails for any reason, the backup is aborted with a `PREPROC_FAILED` error.
`N/A`	`snapproc.bat`	Allows Windows® clients an extra processing step between snapshot creation and backup. If this script fails, the backup is aborted with `SNAP_FAILED` error and an attempt is made to perform post processing.
`postproc`	`postproc.bat`	Used for post phase backup processing, such as cleaning up a snapshot or starting up an application that was shutdown in the pre-process phase. This script is called once the backup is attempted, regardless of the actual backup status. The error code for the post-processing phase is only reported to the server if the backup completes successfully. It is recommended that this phase should be written to send e-mail or some other form of alert if there are problems.

All scripts take exactly two arguments. The first argument is the type of backup performed and is either "full" or "incr". The second argument is the pathname of the file system, directory, or single file that is being backed up. Therefore, scripts can be written to only take action on selected backups, while performing no useful function on other client backup volumes. Example scripts are located in the examples/client directory where your TiBS server is installed.

AFS® Backups

There are several steps involved in processing AFS® volumes and **the order is important**.

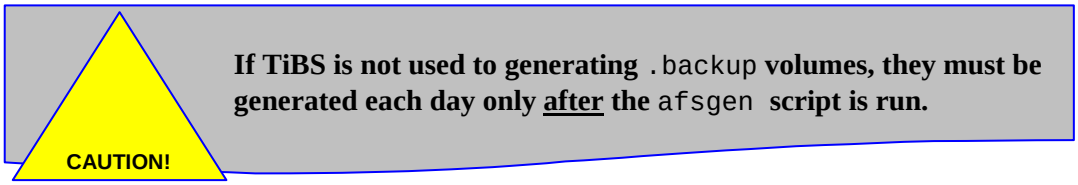
1. Generate the current backup definitions for the AFS® cell.
2. Generate `.backup` volumes for volumes that changed since the last backup.
3. Run backups to the backup cache and let `tibstaped` write them to tape.

Updating Backup Volume Definitions

The `afsgen` script updates definitions, typically each day, before backups run. It detects volume creation, deletion, and relocation. It is possible to modify an AFS® volume between the time when the script runs and the time when backups run. In these cases, it may take an extra day for the backup server to "catch up" with these changes. To minimize this problem, schedule backup functions during times when cell administration and cell activity is low.

Generating .backup Volumes

The `.backup` volumes are automatically updated during the processing of full network backups. TiBS can generate them as part of normal processing of incremental backups by enabling `AFS_AUTO_GEN` in `tibs.conf`.


CAUTION!
If TiBS is not used to generating .backup volumes, they must be generated each day only after the afsgen script is run.

Running Backups

To run multiple fileserver backups in parallel, use the **`afsbackup`** script. This script will prompt for an admin token (if one is needed) and process any volumes that have not been written to the media pool. The definition of the pool determines the type of backup performed.

```
# afsbackup -l pool < additional arguments>
```

To backup an individual fileserver, use the **`-s fileserver`** option to **`afsback`**.

```
# afsback -l pool -s fileserver
```

To backup an individual AFS® volume, use the **`-n volume`** argument to `afsback`:

```
# afsback -l pool -n volume
```

Note: **By default, backup operations will complete on all AFS cells that are valid for a given tape pool. If your site supports multiple AFS cells, you may need to specify the cell that you are working on with the `-c cell` option.**

AFS® Volume Time Stamp Considerations

To keep track of what volumes need a backup, the backup server keeps track of each volume's Last Update Time as a string in `YYYYMMDDhhmmss` format. With TiBS, volumes that have not changed since their last backup will not be backed up again. To ensure that these volumes still appear on the most recent incremental tape, use the **`-F`** flag to `afsback` or `afsbackup`. This option will consume more tape resources, but keep all of the current volume updates on a single media pool.

Backup of Additional Fileserver Directories

For disaster recovery of a fileserver, define the following volumes as client volumes for each fileserver:

```
fileserver|/usr/afs/etc|none
fileserver|/usr/afs/local|none
```

This will backup the minimum configuration information for the fileserver. Recovery will require a reinstallation of the fileserver product.

Backup of Additional Database Server Directories

For disaster recovery of a database server, define the following volumes as client volumes for each database server:

```
dbserver|/usr/afs/db|none
dbserver|/usr/afs/etc|none
dbserver|/usr/afs/local|none
dbserver|/usr/vice/etc/CellServDB|file
dbserver|/etc/rc/afs|none
```

This will backup the minimum configuration information for the database server. Recovery will require a reinstallation of the operating system and the database server product.

Automating AFS® Backups

TiBS now has the ability to completely automate backup processing for AFS. Install the AFS fileserver product on the backup server and incorporating it into the cell. TiBS is able to run AFS commands such as "`vos dump`" with the `-localauth` option, which does not require an admin token. The backup server does not need to have any vice partitions and can act as a "dummy" AFS fileserver. The `-localauth` option to AFS commands is enabled by setting `VOS_LOCALAUTH` in `tibs.conf` to 1.

Expiring Backup Volumes from Media Pools

Individual Clients and Volumes

Operators may wish to replace the current backup volume(s) within a media pool for a given client. Two good reasons for this are:

A client's volume fails and a restore is required. A full backup of the restored volume is now required.

The client that the data volume resides on is being shutdown and a final incremental backup is required to today's cumulative or true incremental media pool.

By default, TiBS will allow only one backup volume for a client volume to be written to a given media pool during a backup cycle. Expiring permits the tape database to place another copy of a given client's backup volume(s) on a media pool that may already have a copy. The old backup volume is still valid for restoring, but it is not part of the current backup for that pool. Use the **`hostexpire`** command:

```
# hostexpire –l pool –n client
```

Or for a specific client volume:

```
# hostexpire –l pool -n client -p path
```

Expiring Older Media Volumes

The `hostexpire` program can be used to expire older tape volumes for scheduling purposes. This allows multiple copies of all client volumes to be written to the same pool over and over again. A site can use a single incremental media pool and use `hostexpire` each day to reschedule backups to the pool each day:

```
# hostexpire –l pool –s seconds
```

Older backup volumes will still be available for restore requests. The program compares the time that each backup volume was written to the media pool with the current time on the backup server. Any volumes found that are older than **seconds** will be expired.

Expiring Multiple Levels of Media Volumes

When more than two backup levels are used it may be necessary to synchronize the timing between the `hostexpire` commands. This can be done by determining the current time in seconds (e.g. `date +%s`) and passing this constant time to each instance of `hostexpire` with the `–f seconds` option:

```
# hostexpire –l pool –s seconds –f seconds
```

Removing Older Media Volumes Permanently

The `hostexpire` program can be used to recycle tapes in a media pool. If tape is used, once all of the volumes on an individual tape have been expired, the tape is free to be re-written. Once backup volumes are expired permanently, they may no longer be accessed for restore requests. Use the **`-r`** flag to permanently remove backup volumes:

```
# hostexpire –l pool –s seconds -r
```

Expiring a Fixed Amount of Data

You can control the amount of data that is expired with the **`-m megabytes`** option. The option limits the amount of data that is expired from the media pool. The amount of data in the cache is not considered. This option may be used to "expire ahead" or to control the amount of data that is expired on any given day. You must add the **`-s 1`** option for the command to succeed:

```
# hostexpire –l pool –s 1 –m megabytes
```

Expiring Only a Disk Library Volume

Disk library backup volumes are removed whenever a corresponding tape volume is removed. In some instances the retention time for the disk library volume may be shorted than the corresponding tape volume. The **`dlremove`** command can be used to remove data from the disk library only. This will only succeed when a media pool has been defined to include disk storage:

```
# dlremove –l pool <additional arguments>
```

Scheduling Backups

Scheduling Concepts

TiBS provides simple yet powerful backup scheduling. Here is a review of some of the key concepts:

Each backup volume is defined to a backup class.
Each backup class defines the valid media pools to which backup volumes in the class can be written.
Media pools may be defined to store data on disk, on tape, or on both.

Media pools contain three types of backup volumes; current, expired, and archived. Any backup volume may be used in a restore process. The current backup volume is the most recent backup of a client volume in the media pool. There may be multiple expired or archived volumes in a media pool. For each pool there are three files:

`entries.txt`	Contains the most recent backup volume for each `host:partition` pair.
`entries.old`	Contains older volumes for each `host:partition` pair. In TiBS full version, this represents the backup volumes that will be scheduled for a backup consolidation.
`entries.arch`	Contains tape volumes that are no longer required for backup consolidation. When using tape, once all of the volumes on a tape have been archived, the tape may be removed off-site.

If a client volume has a current backup volume (in `entries.txt`) in a media pool then no backup to the pool is required. The existence of a backup Tag within the `entries.txt` file also means that no backup to the media pool is required at this time. Backup Tags are typically used in multiple level backup strategies and can also be used to schedule backups to balance the backup load.

Managing Media Pools

If a client volume does not have a current tape volume or appropriate Tag in `entries.txt` for a given media pool, then a backup request is outstanding for the client volume to that pool.

Current backup volumes can be expired automatically with the `hostexpire` command:

```
# hostexpire –l pool –D days -y
```

The time in days may be replaced with the **`–s seconds`** option for maximum scheduling flexibility. All expiration is based on the time that the volume was written to the media pool (not the time that the volume was backed up to the cache). Any volumes that are at least `days` (or `seconds`) old will become expired. The data is still valid, but a new backup will be sent to the pool if requested.

Current or expired tape volumes can be removed from the media pool permanently if they are not being retained permanently. This is typically done with daily or weekly incremental data:

```
# hostexpire –l pool –D days –y -r
```

`ThisDaily.txt` is a file located in each class directory that contains the current cumulative or true incremental media pool. This is used by roaming clients and must be updated regularly if more than one media pool is used by the class for daily network backups.

`ThisFull.txt` is a file located in each class directory that contains the current full media pool. This is used by roaming clients for new backups (Full and Lite versions) or when a new backup is needed (TiBS Small Business Editions). This file must be updated if more that one full media pool is used by the class for full network backups.

Timing Considerations for Media Expiration

When calculating the number of seconds or days to send to `hostexpire`, subtract one day from the desired expiration result. For example, if the tape volumes for a given pool are to be expired once each week, the number of days used in the calculation should be 6, not 7. This is because the actual tape volumes that were written one week ago have time stamps between 6 and 7 days old, not greater than 7 days old.

This same principle applies to interdependencies among tag pools. For example, it is better to expire a tag "weekly" pool every 6 days and its parent "monthly" pool every 27 days instead of every 28. Failure to properly expire tag pools can result in extra work load on the backup server.

Automating Backups

The `netbackup` program is a useful way to schedule and control network backups. The `-P` process option controls how many backups should be run in parallel. This value can be set up to the number of valid backup processes you site has licenses for on a given backup server.

```
# netbackup -l pool -P process_count
```

The `mergebackup` example script will perform any outstanding tape merges for a given pool easily:

```
# mergebackup -l pool -n process_count
```

When using tapes for a media pool, the `process_count` should not exceed the total number tape drives that are available to the media pool to mount tapes for read access.

There are some order requirements when using `teramerge` on midlevel backp pools (see Using Tags to Implement Multi-level Backup Strategies). The merge on any tag pool must be run before a given pool is merged (see Example 3: Advanced Scheduling).

Example 1: TiBS Small Business Edition

This example implements a simple 2-level backup

```
#!/bin/sh
# sbe_example.sh: simple backup example
#
# 1. new full backup each week.
# 2. new incremental backup each day.
# 3. Recycle daily backups after one week.
# 4. Keep full backups permanently.
#
# get the tibs scripting environment
. /etc/tibs.conf

# schedule weekly backups (note: n-1 rule applies)
hostexpire -l Full -D 6 -y

# schedule daily backups (trick: 1 second expires the Incr pool)
hostexpire -l Incr -s 1 -y

# permanently remove daily backups older than 1 week
hostexpire -l Incr -D 7 -y -r

# run full network backups, wait for all backups to complete
netbackup -l Full –P 10

# run daily backups, wait for all backups to complete
netbackup -l Incr –P 10

# report for all TERA_ADMINS and REPORT_ADMINS defined in tibs.conf
genreport -a
exit 0;
```

Example 2: TiBS Lite Version

In this example the `labels.txt` file would be:

```
Full|Weekly|none:unused|none|full|28|0|1|1|archive_full
Incr|none|none:unused|none|incr|1|28|1|1|active
```

And the automation script is:

```
#!/bin/sh
# lite_example.sh: 2 level backup example for TiBS Lite Version
#
# 1. new full backup each month.
# 2. new incremental backup each day.
# 3. Recycle daily backups after 4 weeks.
# 4. Keep full backups permanently.
#
. /etc/tibs.conf

# schedule backups consistently
hostexpire -l Full -D 27 -y
hostexpire -l Incr -s 1 -y

# permanently remove daily backups
hostexpire -l Incr -D 28 -y -r

# run network backups (run full for new hosts)
netbackup -l Full –P 10
netbackup -l Incr –P 10

# run merge backups
mergebackup -l Full -w

# report for all TERA_ADMINS and REPORT_ADMINS defined tibs.conf

genreport -a

exit 0;
```

Example 3: Advanced Scheduling in TiBS Full Version

In this example the `labels.txt` file would be:

```
Full|Monthly|none:unused|none|full|364|0|1|1|archive_full
Monthly|Weekly|none:unused|none|flush|28|0|1|1|archive_full
Weekly|TIncr|none:unused|none|flush|7|84|1|1|active
TIncr|none|none:disk|none|tincr|28|0|1|28|active
```

And the automation script is:

```
#!/bin/sh
# four_level.sh: 4 level backup example for TiBS Full Version
#
# 1. new full backup each year (actually every 364 days).
# 2. new cumulative incremental volume each month (actually every 4
weeks).
# 3. new cumulative incremental volume each week.
# 4. new true incremental backup each day (also stored in disk library)
# 5. Recycle daily backups after 4 weeks.
# 6. Recycle weekly merge backups after 12 weeks.
# 7. Keep full and monthly backups permanently.
#
. /etc/tibs.conf

NOW=`${TIBS_HOME}/bin/tibstime`

# schedule backups
hostexpire –l Full –D 363 –f ${NOW} –y
hostexpire –l Monthly –D 27 –f ${NOW} –y
hostexpire –l Weekly –D 6 –f ${NOW} -y
hostexpire –l Incr –s 1 -y

# permanently remove weekly and daily backups
hostexpire –l Weekly –D 84 –y –r
hostexpire –l Incr –D 28 –y –r

# run network backups (run full for new hosts)
netbackup –l Full –P 10
netbackup –l Incr –P 10

# run merge backups (note, the order is important)
mergebackup –l Weekly -w
mergebackup –l Monthly -w
mergebackup –l Full -w

genreport –a

exit 0;
```

Example 4: Production Script: mysiteauto

TiBS now ships with an example automation script designed to work for all backup schedules (up to 4 backup levels) and versions of TiBS (Full, Lite, or Small Business Edition). This script has several settings that are configurable for backup schedules, tape retention policies, enabling AFS backups, etc. There is also a TESTING mode that shows the TiBS commands that will be run based on the current settings. The output will look similar to the commands listed in the above examples. You can use the output from TESTING mode to manually execute the commands in sequence to gain a better insight into how TiBS works.

Restore Procedures

You may restore data from any volume to any directory on any client running the client software. Be aware that some meta data will not be restored if the target restore client does not support it. For example, Windows® NT file security information will not restore to a UNIX® client. Refer to the Command Reference for additional options to the **tibsrest** program:

```
# tibsrest –n client –p path <additional arguments>
```

During the restore process the program will prompt for any tapes that are not already mounted:

> **TIP: The `-q` option to find out what tapes are needed to process a restore request. Once the tapes are mounted run the program again without the `-q` to complete the restore.**

Restoring To an Alternate Location

To restore data to an alternate backup client and/or restore path use the **`–r althost`** and/or **`–a altpath`** options to `tibsrest`:

```
# tibsrest –n client –p path –r althost –a altpath
```

Restoring Older Backup Volumes

By default, `tibsrest` will restore data from the most recent backup. To restore older data use the **`–t timestamp`** option. The program will accept any subset of the time string as long as it includes `YYYY.`

```
# tibsrest –n client –p path –t YYYY/MM/DD/hh/mm/ss
```

Incremental Restores

To restore incremental data use the **`–i`** option to `tibsrest`. The program will only read data from the most recent incremental backup volume that it finds. This is useful when performing a single file restore to prevent the unnecessary reading of full or lower level tape volumes. Data located on the most recent backup volume comes directly from the backup cache.

```
# tibsrest –n client –p path -i
```

Single File or Sub-Directory Restore

In order to avoid searching through tape volumes manually, backup administrators can use the **`filesearch`** utility to locate the proper file version quickly and easily. This utility will scan all matching file lookup databases for the requested file or directory string. It is used to obtain the parameters for the **`-t timestamp`** and **`-s subdir`** options to `tibsrest`. The only required argument is a search string:

```
# filesearch -s string
```

Where `string` is at most, the pathname of a file or directory relative to the backup volume path. For example, if the file that is requested is `/home/user/lostfile`, and was backed up on `/home`, a valid `string` would be `user/lostfile` or any other subset string (e.g. `lostfile`).

Note: Without additional arguments this command will search every file lookup database in the tape database.

Limit the Search

If the name of the client where the file originally resided is known, **limit the search by fully qualified hostname**:

```
# filesearch -s string -n client
```

If the partition the file originally resided is known, **limit the search by backup volume**:

```
# filesearch -s string -n client -p path
```

If the exact path or filename of the file is known, enable the **case sensitive search**:

```
# filesearch -e -s String -n client -p path
```

If the requested data is a subdirectory of a client partition, enable **directory only matching**:

```
# filesearch -d -s string -n client -p path
```

If the requested data is a single file, enable **file only matching**:

```
# filesearch -f -s string -n client -p path
```

TIP: The time to search and the amount of output produced depends on how many tape backup volumes match the arguments passed to `filesearch`. Count the searches without actually performing them with the `-c` flag.

Example Filesearch Output

```
bash-2.05a# filesearch -n host -p /home -s user/.login
SEARCHING PATH /home
SEARCHING STRING user/.login
filesearch will need to scan 28 incr fldb records
filesearch will need to scan 62 flush fldb records
filesearch will need to scan 15 full fldb records
Begin incremental search
Begin flush search
############ FLUSH TAPE Month.157 offset 249 date 2003/10/23/20/11/29
FILE               374 Wed Oct 23 17:01:57 1996  host|/home|/user/.login
Begin full search
############# FULL TAPE Full.170 offset 13 date 2003/10/23/20/11/29
FILE               374 Wed Oct 23 17:01:57 1996  host|/home|/user/.login
```

The script first determines which tape databases to scan based on the `hostname` and `path` arguments. It then runs `fldblist` and uses the string matching rules to print the relevant results. Note that the date field in `YYYY/MM/DD/hh/mm/ss` format can be used to select the file that needs to be restored with `tibsrest` from the most recent tape (in this case `Apr.00.3`):

```
# tibsrest -i –n hostname –p /home –s user/.login –t 2003/10/23/20/11/29
```

In this example, the timestamp on each file is the same, so any restore of the file will yield the same result.

Windows® Recovery Procedures

Full system recovery is available for Windows® NT/2000/XP/2003 systems with the Open Transaction Manager™ (OTM) option for TiBS. For Windows® 95/98 or sites that do not run software that supports open files, full system drive recovery is not supported. If you intend to use TiBS to recover Windows® system drives, it is highly recommended that you purchase the OTM option for TiBS.

Recovering a System Drive

Mount the replacement drive onto a Windows® system that is running the TiBS backup client to an alternate mount point (e.g. `F:`). *To ensure a proper system recovery restore to a system running the same operating system as the failed system drive*. Format the new drive for the appropriate file system (e.g. NTFS). Restore the data with the `tibsrest` command to the new partition:

```
# tibsrest –n failed_host –p /c –r restore_host –a /f
```

Windows® Registry Edit

The old drive identity in the registry must be deleted before booting the new partition. This can be done by using `regedit.exe` to import the registry into the system where the restore was performed. For example, if the system was restored to `F:`

```
C:\> regedt32 (or regedit on some XP systems)
```
Select `Registry -> Load Hive`

Use the browser to select `F:\winnt\system32\config\system`

You will be prompted for a key name to mount the hive (e.g. `MY_MOUNT`)

This will open the restored registry on the current system as a subkey. From the subkey, locate:

```
MY_MOUNT\CurrentControlSet\MountedDevices

OR

MY_MOUNT\MountedDevices
```

If the system was originally mounted as `C:` then delete the key `\DosDevices\C:`. This will clear the `C:` device for boot of the newly restored system drive. Exit `regedit.exe` to save the modified registry file.

Booting the Recovered System

Once the restore completes, mount the new disk in the failed system and use a boot floppy to set it as an active partition if necessary. Boot the recovered system.

Recovering a Windows® Registry

Take the repair diskette and the three Windows® NT installation diskettes (or an installation CD if the system can boot from CD) to the failed system. Mount the new disk drive and insert the first Windows® NT installation disk into the bootable floppy drive (or mount the install CD). Turn the machine on. Insert disk 2 when/if it is requested. When prompted, select the option to Repair by pressing the R key. Select the options as follows, (OTM users do not need to select Inspect Registry Files,):

[X] Inspect Registry Files
[X] Inspect Startup Environment
[ ] Verify Windows NT System Files
[X] Inspect Boot Sector

In the Inspect Registry Files menu, select all the registry keys for restore.

[X] SYSTEM (System Configuration)
[X] SOFTWARE (Software Information)
[X] DEFAULT (Default User Profile)
[X] NTUSER.DAT (New User Profile)
[X] SECURITY (Security Policy) and SAM (User Accounts Database)

Once the repair process is complete, remove the repair disk and restart. The Windows® NT system partition should start normally.

AFS® Restores

Restore of AFS® volumes is performed back to a fileserver with the **`afsrest`** program. A new AFS® volume with a **`.restore`** extension to the original volume name, at the last known location of the original volume is created:

```
# afsrest –n volume < additional arguments>
```

By default, restore fails if the volume already exits. The **`–o`** flag is used to overwrite the original volume name, and the **`–f`** flag is used to overwrite any existing volume that you are trying to restore.

TIP: **The `-q` option to find out what tapes are needed to process a restore request. Once the tapes are mounted run the program again without the `-q` to complete the restore.**

Note: **If your site supports multiple AFS cells you must specify the cell that you are working on with the `–c cell` option.**

Restoring Incremental Data

Single volume or sub-directory restore is currently unsupported for AFS® volumes. You can still restore data from a single incremental backup volume with the **`–i`** option to `afsrest`. The restore will not require a full tape mount, and will not require any tape mount if the data can be retrieved from the backup cache:

```
# afsrest –n volume –i
```

Restoring to an Alternate Location

Redirect data to an alternate volume name, vice-partition, or fileserver by adding any combination of additional arguments:

```
# afsrest –n volume –a newvol –p newpart –r newserver
```

Restoring to a Specific Time

To restore an AFS® volume to a certain point in time, use the `–t` option to `afsrest`. The minimum acceptable time string is `YYYY`.

```
# afsrest –n volume –t YYYY/MM/DD/hh/mm/ss
```

Restore of AFS .readonly Volumes

Support for restore of AFS readonly volumes is still under development. At this time, you must restore to an alternate volume name (possibly the read-write volume). You may want to rename the current read-write volume, perform the restore, release the volume, and then move the current read-write volume back to its original name.

Disaster Recovery for AFS®

To recovery an entire AFS® cell, fileserver, or vice-partition from tape, first query the tape database with the **`afsrecover`** command:

```
# afsrecover -s server -p partition -q
```

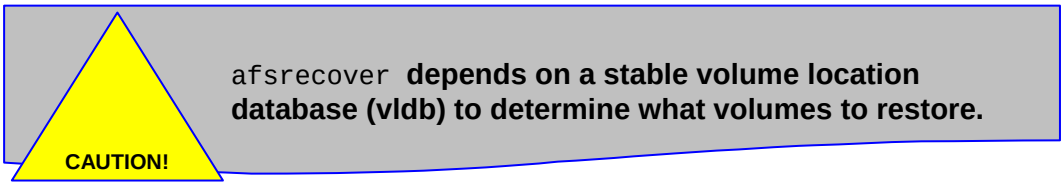
The command will list all disk and tape backup volumes in the order that they will need to be read to complete the operation. It is possible to find older backup volumes when running a query. You can limit the full media pool with the **`-u pool`** option:

```
# afsrecover -s server -p partition -q -u pool
```

Once you have determined which tapes (if any) will be read, locate all of the tapes you will need to process the recovery. You can redirect data to an alternate fileserver, an alternate vice-partition, and/or alternate volume names with additional parameters:

```
# afsrecover -s server -p partition -r restserver –a restpart –e volext
```

You cannot redirect an entire fileserver to a single partition. You can redirect an entire fileserver to a new fileserver, as long as the vice-partitions on the new fileserver are identical to the old one and have at least as much free space.



Note: **If your site supports multiple AFS® cells you must specify the cell that you are working on with the `-c cell` option.**

Recovering an AFS® Fileserver

Recovery of an AFS® fileserver requires some disaster recovery planning. The basic strategy is to have the following procedures ready and documented:

Recover the base operating system: This usually involves the installation CD and some pre-set requirements for your site, such as partitioning schemes and add-on packages.

Re-install of the fileserver server product: This step may not be necessary if the appropriate data is backed up using TiBS.

Recover server configuration files: Accomplished by restoring the necessary directories for the fileserver configuration from TiBS. At this point the fileserver may be started with **`bos`**, or rebooted.

Recover vice-partitions: Accomplished by rebuilding any necessary vice-partitions and restoring backed up volumes using `afsrecover.`

Recovering an AFS® Database Server

Recovery of a database server requires some disaster recovery planning. The basic strategy is to have the following procedures ready and documented:

Recover the base operating system: This usually involves the installation CD and some pre-set requirements for your site, such as partitioning schemes and add-on packages.

Re-install the database server product: This will not be necessary if the appropriate data is backed up using TiBS.

Recover database files: Accomplished by restoring necessary directories for the database server from TiBS.

Once all data for the database server has been recovered, start the server with `bos` or reboot the machine.

Backup Monitoring

Monitoring Backup Server Activity

The **tibstat** command is a basic monitoring tool that allows backup administrators to monitor the progress of backup and restore processing. With no arguments, all activity is listed as follows:

Pending Volumes:	The list of backup volumes that are waiting to be written to tape. Use the **-p** flag to report just the pending volumes.
Busy Volumes:	The list of backup volumes that are currently being processed. Use the **-b** flag to list just the busy volumes.
Processes:	A list of TiBS processes that are currently running. Use the **-j** flag to list just the currently running processes.

To see a combination status, use more than one flag. For example, to see busy volumes and running processes, run:

```
# tibstat –b –j
```

Backup Report Generation

Use **genreport** to monitor daily progress of the backup server and alert administrators of potential problems. This command summarizes all of the backup server log files and statistics into a concise and informative daily report. The report can be run automatically or manually each day. Both backup administrators (TERA_ADMINS) and report administrators (REPORT_ADMINS) can be configured in tibs.conf to receive this report each time it is generated. To send an entire report to all administrators run:

```
# genreport –a
```

Filtering Reports

Standard reports may contain more information than is required for your site. You can filter specific files used in generating reports by adding a unique sub-string to the appropriate **file.flt** filter file in the reports directory. For example, to filter out messages from the tapelog.txt report, add a unique sub-string to the tapelog.flt file.

Customizing Reports

Some sites may not need all of the information offered by the full report each day. In this case, we recommend creating a script that calls genreport with all of the flags required for normal report generation. See the Command Reference for a list of the flags available for custom reporting.

Cache Statistics

Cache statistics help to optimize or track backup server performance. The **`cachestat`** script will analyze the backup cache and provide information on:

Network activity:	Client volumes that use the most network resources, typically because they are generating or updating the most data.
Cache activity:	Client volumes reusing the most files (by total size) from the backup cache. These volumes may be candidates for consolidation to free needed cache resources.
Size:	The largest client volumes in the backup cache. These volumes may be candidates for consolidation to free needed cache resources.
Aggregate activity:	The totals for all network and backup cache usage based on the last successful incremental backup for each client volume. This statistic reflects the success of TeraMerge® in reducing the network load for the backup function.

Tape Statistics

Tape statistics give backup administrators useful information about how data is being stored on tape. To determine the amount of data stored in a media pool use the **`tapestat`** command:

```
# tapestat –l pool
```

To view cache and network statistics for cumulative incremental backups use the **`–n`** option:

```
# tapestat –l pool -n
```

This option will also summarize the top twenty backup volumes in cache utilization order. This information can help determine the best backup volumes to consolidate from the cache to free needed space.

To view a listing of the largest client files stored in a media pool, use the **`bigfiles`** command:

```
# bigfiles –l pool –c count
```

This command will run `fldblist` on all backup volumes currently on tape for the `pool` and summarize the largest files it finds.

Error Recovery

Location of Backup Server Log Files

All audit and backup activity log files are stored in the **`reports`** directory located in the backup server's installation directory. The standard logs in this directory are:

`cachemgr.txt`	File contains log information for `cachemgr`.
`classupdates.txt`	File contains updates from `hostadd` and `hostdel`.
`disklog.txt`	File contains messages generate by the `tibsdld`
`rlm.txt`	Reprise License Manager™ (RLM) log file.
`mounter.txt`	File contains summary information about program mount requests.
`netbackup.txt`	File contains messages generated by the `netbackup` program
`notify.txt`	File contains summary information about program notification requests.
`tapemgr.txt`	File contains log information for `tapemgr`.
`tapelog.txt`	File contains status information for `tibstaped`.
`tibswd.txt`	File contains log information for `tibswd`.

More detailed information about individual clients is located in the **`reports/clients`** sub-directory.

Note: **The `genreport` script uses these files when creating reports. Each time the script is run it appends the contents of each file to a running log in the `reports/archive` directory.**

Backup Server Failures

Below are the documented backup server failure modes and recovery procedures for each. If you are seeing other types of failures or an UNDOCUMENTED failure and cannot resolve the problem, please contact Teradactyl® technical support.

ACCEPT FAILED

The client connection was initiated but is never completed. The most common occurrence of this error is with improperly configured Windows® clients. Check the network configuration for the client against the class definition for discrepancies.

AFSSRV_ERROR

This occurs when an AFS® backup or cache recovery inconsistency is found. More information on the type of failure can be found in `reports/clients/afslog.txt`.

AFS_ERROR

An AFS® command request such as `vos` failed. In many cases, this type of error will require the attention of a system administrator. In some cases running a *`bos salvage`* on the read-write volume and regenerating the backup volume will correct the problem.

AUDIT_CLIENT

This occurs when a backup client cannot process a request from `hostaudit`. Refer to the clients `teralog.txt` file for more detailed information

AUTHENTICATION

The client has not yet been authenticated with the server. You can authenticate a client from the server with the `hostaudit` command and retry the operation.

BAD_CHECKSUM

The checksum on a file failed during a transfer. Typically indicates corrupted media (disk or tape). This error can also be generated by faulty networking hardware between the backup client and backup server.

BAD_DEVICE (Windows® only)

An unrecognizable or unsupported file system was found when trying to backup a Windows server or workstation.

BAD_DIRECTORY

The client failed to read all of the contents of the reported directory. In this case, the backup is aborted. Administrative intervention may be required to determine the cause of the failed directory access if this error persists. This can also happen on Windows® disk partitions that are completely empty. In this case, the error is the result of a failure to lookup any file on the empty directory.

CACHE_FULL

The cache volume could not get enough space to complete the operation. If you have more than one cache available enable cache balancing or manually move volumes to gain the needed space.

CACHE_NOT_FOUND

This happens during an incremental backup if there is no current backup for a client volume. Once a current full backup is available, then incremental backup processing may continue.

CACHE_PARAM

The configuration file, `caches.txt`, in the backup server's state directory may be corrupted. Shutdown the server with `stoptibs` and check that `caches.txt` is properly configured (see Defining the Backup Cache)

CHECKAFS_ERROR

Reports that a newly generated TiBS backup volume for AFS was not created properly. Only occurs when using the check flag for AFS backup commands.

CLIENT_ARGS

A remote backup client did not communicate properly with the backup server. Check the client commands and the log file for the client on the server located in `reports/clients`.

CLIENT_CONFIG

The backup client's `tibs.ini` file contains an unexpected hostname. Typically, this is due to a wrong `hostname` being entered during the install process. Once the client's `tibs.ini` file is corrected, remove the `client.auth` and `client.dat` security files on the client and re-authenticate the client from the server:

```
# hostaudit -n host.domain -r
```

CLIENT_REFUSED

This error indicates that the backup server was unable to establish a TCP/IP connection with the client. This may indicate that the client has not been set up properly or has hung.

CONNECT_FAILED

An unexpected error occurred during a network connection. This may indicate that the client has not been set up properly or has hung.

DRIVE_IDENT

The device identifier for the client volume has changed because the device has been replaced. TiBS cannot continue to backup this device incrementally. To take a new full backup, expire the volume from the tape database and disk cache:

```
# hostexpire –l full_pool –n host –p path –x -y
```

EMPTY_DRIVE (Windows® only)

A file not found error occurred when trying to open a disk partition. Check the client configuration to make sure that the partition is still valid.

ERROR_DUMP

A client was unable to send an error log file to the server. You may need to manually connect to the client to view the contents of the error log.

FLUSH_VOLUME
The server was unable to open the cache database (`full.tdb`) for the client volume during a backup consolidation operation. If the cache directory has been removed, use the `cacherecover` program to restore the cache to the most recent state.

HOST_UNREACHEABLE
The TiBS backup server could not find a route to the host. Typically, the backup client is down or off the network. Retry the operation once network connectivity is re-established.

HOST_LOOKUP
This error indicates that the backup server was unable to resolve the IP address of the client. This error is usually caused by a bad hostname configuration or a host removed from service. This type of error is usually resolved by removing the client volume definition(s). (see Removing Client Definitions).

INCOMPATIBLE
The client software is not compatible with the current server revision. This can happen during certain upgrades, when the client/server interaction is no longer compatible with a client machine that missed the upgrade. Update the client software and retry the backup. Alternatively, you can add the client version, "TiBS 2.x.x.x" to the file `state/versions.txt` to allow the server to continue to communicate with the client.

INCR_PENDING
This error is caused when a merge process is waiting for a network incremental backup to succeed before continuing. This happens when the merge processes is being recovered using `cacherecover`. The merge will succeed once a successful incremental backup has been taken.

INVAL_PARTITION (Windows® only)
The partition that has been defined for backup is invalid or unsupported.

KILLED
The process was killed by a signal from the system or by manual intervention. The operation failed. Retry the operation.

LICENSE
No valid license was found. Make sure that the license daemon was installed properly and that the license file is valid. Contact Teradactyl® if you require additional Operating System licenses or your current license file has expired.

MERGE_PENDING
A tape merge is in processes. No other backups on a client volume may be performed until the specific merge process finishes. This is done to keep the redundancy and tape failure recovery requirements in TiBS.

MERGE_STATE
An unexpected state occurred during the merge process. This is typically caused when a tape volume fails and the server is unable to resolve the proper tape volumes to continue (see Recovering from a Failed Tape Merge).

MISSING_FILES
The wrong file version was found during a merge process. The backup volume may be corrupt. If you see this message, contact Teradactyl® immediately.

NETWORK_READ

A network connection was dropped by the remote host. Check the remote hosts log file for additional information.

NETWORK_TIMEOUT

This means that the backup server was unable to reach the client over the network. Once the network and client connectivity is re-established, retry the operation.

NO_AUTH

The client is not properly authenticated with the server. Check the client and server authentication files and reauthorize.

NO_DEFINITIONS

The server failed to find any class definitions based on the command line given. Typically, this may be caused by a new host that has not yet been defined to TiBS. Use the `hostaudit -l` command to view the current definitions for the client and the `hostadd` command to add new pathnames as needed.

NO_DEVICE (Windows® only)

Typically indicates a CD-ROM, floppy disk, or other removable device. Check the client drive configuration.

NO_FULLDUMP

This happens during an incremental backup if there is no current backup for a client volume. Once a current full backup is available, then incremental backup processing may continue.

NO_PARTITION

The client could not find the pathname specified by the server. Check the client configuration in `state/classname/groupname.txt` for errors. You can also audit the client with the `hostaudit` command.

NO_SECOND_FULL (TiBS Full Version ONLY)

A backup volume has been scheduled for a full tape merge and a network full backup was attempted. The network full backup is aborted. The error will be corrected once the new full tape volume is produced by the merge process.

NO_TAPE_FOUND

A process failed to open a drive that was mounted for use. Check the drive device to make sure the media was inserted properly and the drive is on-line. Dismount and remount the drive and retry the operation.

OPER_ABORT

A manual operation failed because the user answered "`no`" to a confirmation request. The job was aborted.

OS_TYPE

This error indicates that the operating system on the client does not match the known operating system on the backup server. Caused when a system changes architecture, or on a multi-operating system machine when data is accessed that is unavailable to the current running operating system. This is typically resolved by configuring the client with `hostadd` and `hostdel`.

OTM_ERROR (Windows® NT/2000/XP/2003 clients only)

The Open Transaction Manager™ failed during backup. The most common cause is due to insufficient cache space. You can increase the amount of cache space by editing `tibs.ini` on the client or use the `hostaudit` command from the server.

OTM_START (Windows® NT/2000/XP/2003 clients only)

The Open Transaction Manager™ failed to start. The two main causes for this error are:

1. There was not enough room on the file system the first time OTM started to obtain a cache file of the specified size. In this case, free up the needed disk space so that the cache can be allocated. Once the cache is allocated it will remain available between backups.

2. The client was sufficiently busy and a quiescent wait period was not established. In most cases, a retry of the backup will yield success. If a particular client continues to fail, it is an indication that something is always running on the system. You may want to check the process table for runaway jobs and/or reboot the system. Another occurrence we have seen is that some screen savers actually prevent OTM from starting up because they are so resource intensive.

3. The client was recently upgraded with a new version of OTM and a reboot may be required.

OTM_TIMEOUT (Windows® NT/2000/XP/2003 clients only)

The Windows client failed to initialize the OTM cache for backup. Use `hostaudit -d` to examine the client's `teralog.txt` file for additional information. This can happen if the OTM cache size is large (several gigabytes) on a slower client.

PATH_NOT_FOUND

The client could not find the pathname specified by the server. Check the client configuration in `state/classname/groupname.txt` for errors. You can also audit the client with the `hostaudit` command.

PREPROC_FAILED

The pre-processing phase for backup failed on the backup client. The backup operation is aborted.

REGISTRY_BACKUP

The TiBS client for Windows® failed to make a copy of the registry for alternate backup. More information about this error can be found in the client's log file.

RETRY_PHASE

The TiBS server has detected that too many files have changes with older modify times. Typically caused by the addition of a large amount of data from another system, or a replaced filesystem.

SERVER_CONF

The server was unable to send backup configuration information to a remote backup request. Check the backup configuration for the client with `hostaudit -l`.

SERVER_STATE

A remote backup request failed because the server was busy or in an unexpected state. Use `tibstat` to view busy or pending volumes. Use `cacheclear` to unlock the state of a cache backup volume if necessary.

SHUTDOWN

A shutdown has occurred. This is provided for information only and typically is not an error.

SIZE_ERROR

The size of a file on the client has changed and is not consistent with the server. A new full backup must be taken to guarantee data integrity. If you see this message, contact Teradactyl® immediately.

SNAP_FAILED

The snapshot phase script of the backup client failed. The backup was aborted.

TAPE_DRIVE_CLEAN

A tape drive cleaning flag has been detected. If tape cleaning is enabled, the drive will be cleaned automatically. Otherwise, a notify request for tape cleaning should be mailed to all `TERA_ADMINS` and manual cleaning is required.

TAPE_DRIVE_ERROR

A tape drive has failed to respond to a tape load request. The drive is typically marked as "failed" and manual user intervention is required. In some cases, a bad tape may cause a tape drive to fail. In this case the tape should be marked as filled and removed from service.

TAPE_FULL

The tape media is full. TiBS now saves state information in the disk cache to continue writing the backup volume across multiple tapes. Mount a new tape labeled for the same pool to continue writing.

TAPE_READ

A tape read error occurred. This may indicate failed media. A retry of the operation will work in some instances. If it is determined that the media has failed, follow the instructions for tape error recovery (see Tape Error Recovery.)

TAPE_READ_FIRST

A tape failed to read the first backup volume. Possibly a sign that the tape device is bad or the tape has failed.

TAPE_RDONLY

A tape write failed, possibly because the physical write protect has been enabled. The tape is marked as full.

UNDOCUMENTED

You can get more information from the server log file for the client located in `reports/clients` or by reviewing the client's local error log file with the `hostaudit` command.

UNKNOWN_OTM

An unknown or undocumented Open Transaction Manager™ error occurred. Consult the backup client's `teralog.txt` file on the server for additional information:

```
# hostaudit -n host.domain -d
```

VOLINFO

The backup client was unable to determine the file system type of the partition where the target directory for backup exists. This may be an unsupported file system or there may be a problem accessing the volume information required to perform a backup.

VOLUME_BUSY

The volume is currently busy. This can happen if a full backup is run concurrently with an incremental backup or a backup request is made on a pending volume. In this case you may need to retry the job that failed once the job that is running has finished.

VOLUME_CORRUPT

A corrupted cache volume was detected during backup processing or writing data to tape. Contact Teradactyl for additional information about how to resolve these errors.

WATCHDOG_TIMEOUT

The server timed out on a hung network connection with a backup client. This can happen when a machine is shutdown during a backup operation. The timeout frees the backup license for use by another backup process.

Backup Volume Errors

The following error and warning messages may occur, even though the backup was successful. These errors may indicate a client system problem and usually require administrative intervention to correct.

ACTIVE FILE

The file time stamp changed during the backup of the file. The entire backup succeeded, but the active file may be corrupt. The server keeps the active status so that the file is automatically retried the next time a backup is taken. Sometimes actives files are log files that may not need to be backed up. You can define the active file to a rule file to remove it from backup entirely in this case.

BAD ACCESS FILE

Reported items are files that were unreadable by the client's backup service and may warrant further attention. Some examples of causes for bad files include stale NFS handles and Windows® virus files.

OPEN FILE

The client backup process could not open the file. This typically occurs on Windows® clients that are not running the Open Transaction Manager™. There is no open file manager for Windows® 95/98/Me.

FAILED FILE

The client failed to backup the full size of the file. This can happen if a file is truncated during the backup process, or if a failed disk read prevented the full reading on the file. In either case, the size of the file is padded with null characters and is marked for retry on the next backup operation. This error can be an indication that a disk is becoming corrupt and administrative intervention may be required.

Locked Cache Volume Recovery

One or more of the cache volumes for backup clients can lock due to incomplete processing of a backup job. This is most likely to happen when the backup server crashes or loses power unexpectedly. To unlock all cache volumes, run the **`cacheclear`** command. You can also clear individual cache volumes while other backups are processing with the **`-n client`** and **`-p path`** options:

```
# cacheclear -n client -p path
```

Backup Cache Recovery

If a disk drive fails, causing some of the available backup cache to go off-line, use the **cacherecover** program to restore the backup cache:

```
# cacherecover –c cachepath
```

If you do not specify a location, the entire backup cache is restored. The most recent copy for each cache volume is determined and read from tape, or from the optional **FAST_CACHE_RECOVERY_DIR**. The order is alphabetical by tape label, then in numerical order by tape number, and finally in order by tape offset. The program will request tapes to be loaded if they are not already online or available from a tape library.

Cache Recovery Options

Use the **–a** flag to relocate data to an alternate cache location:

```
# cacherecover –c cachepath –a alternate_cachepath
```

If an individual cache volume is deleted accidentally or has become corrupted, it may be restored with:

```
# cacherecover –n client –p path
```

To restore an individual AFS® backup volume to the cache, use the special hostname **afs**:

```
# cacherecover –n afs –p volume
```

For more options see the `cacherecover` command line documentation.

Cache Recovery Failures

Cache recovery may fail on an individual volume. The following are some workarounds for certain types of recovery failures:

1. If the `FAST_CACHE_RECOVERY_DIR` option is enabled, but the data is not found in the recovery database, force the recovery from tape with the **–x** option:

   ```
   # cacherecover –c cachepath –n host –p path -x
   ```

2. When using tape, if the tape volume is determined to be unreadable and the media pool is mirrored, manually dismount the failed tape and mount the alternate tape for read. If the media pool is not mirrored, mark the failed tape volume in the tape database as bad (see **Step 2** in Recover from Previous Tapes).

Once the current failure has been resolved, restart the recovery with the **–R** flag:

```
# cacherecover –c cachepath –R
```

Disk Library Recovery

Backup volumes in the disk library may be recovered from tape using the **`dlrecover`** command. The disk library is represented as a mirror to data on tape. Recovery may be performed most efficiently by identified the recent tapes needed and performing the recovery in parallel for each tape:

```
# dlrecover -l pool -t tapeno
```

In some instances, only data needed for tape merging going forward is require. Use the **`-m`** option to recover only backup volumes needed for merging back to the disk library.

Server Lock File Recovery

TiBS programs use simple file locking mechanisms to ensure that programs read and modify the backup server database *atomically*. All locks are created in the **`locks`** directory. Two lock files are reserved and should not be deleted manually:

`lockdb0:`	Used to access class definitions
`lockdb1:`	Used to access tape drives
`lockdb2:`	Used to control tape libraries and tape mounting procedures
`lockdb3:`	Used by Kerberos version of TiBS (U.S. Only)
`lockdb4`	Used to control access to the pending backup directory
`lockdb5:`	Reserved by the `cachemgr` to inform `tibswd` that it is still running
`lockdb6:`	Reserved by the `tapemgr` to inform `tibswd` that it is still running

If a lock file is not removed by a process, all operations that require access to the lock will hang. You should receive the alert: "`tibswd alert: lockdbX may be hung`"

To remove a lock file, view the process ID that last held the lock from the lock file itself and make sure that the process is no longer running. If the process ID is not present or zero, then the last process to use the lock required read-only access. Remove the lock file to re-enable processing on the backup server.

Backup Server Recovery

To recover a backup server from media requires careful disaster recovery planning. To implement a disaster recovery plan, document all procedures. Make sure all electronic information is available in a removable media form for an off-site location. The current process for completely restoring/rebuilding a backup server from removable media includes:

- Re-install the base operating system and standard local software from CD-ROM or other installation media.

- Re-install TiBS from CD-ROM and perform the following minimal configuration tasks.

  Update `state/tibs.conf` for the site. It is recommended that a hard copy of this file be kept with offsite recovery information including install software.

  Redefine the tape drives and optional tape libraries available for the recovery in `state/drives.txt.`

  Redefine the backup caches in `state/caches.txt`.

  Define a recovery class and a recovery group. You can use any name for the class and group.

  Define labels for the tapes that will be used in the recovery process in `state/classes/recovery/labels.txt`.

- Mount the last tapes written. To do this you need to manually create the `state/barcodes.txt` entry in the format:

  barcode|pool.tapeno|active

  If any of the tapes you are scanning are mirrored, then you need to create entries for both tapes.

- Run `tapescan` to recover the tape database for that tape. If the recovery volume that you need is not there, you may need to scan additional tapes.

  ```
  # tapescan –b –l pool –t tapeno
  ```

- You now need to perform a series of restores that will bring the server database back completely.

  ```
  # tibsrest -n backup_server -p /usr/tibs
  ```

- Perform cache and disk library recovery procedures for any lost backup cache partitions (see Backup Cache Recovery).

Tape Error Recovery

Recovering a Lost Tape Database

If a tape or pool of tapes is improperly removed or recycled the tape database can be recovered in one of two ways:

By restoring the subdirectory for the media pool from the disaster recovery tapes:

```
# tibsrest –n server –p /usr/tibs –s state/tapes/incr/pool
```

By scanning each tape individually:

```
# tapescan –t device -b
```

Removing Unreadable Tapes from the Tape Database

If an entire tape had been destroyed and is unreadable, just delete the tape:

```
# tapedel -l pool -n bad_tapenumber
```

Answer *yes* when prompted to clean up the file lookup database.

For A Single Failed Restore Volume

If the volume is part of a mirrored set, retry the restore with the alternate tape first. If a single tape volume is determined to be unreadable, make a note of the media pool and type, the tape number, and the tape offset. Use the `hostexpire` program to permanently remove the specific tape volume:

```
# hostexpire –l pool –n client –p path –n tape_number –o tape_offset
```

Re-run the restore, if an alternate restore path can be found, the restore program will request the new set of tapes to complete the request.

If an alternate restore path cannot be found, use the **`-x`** option to `tibsrest`. This will allow the restore program to use the closest available set of tape volumes to run the restore. Upon completion a complete list of the files that were not restored will be displayed.

Recovering from a Failed Tape Merge

Recover from previous tapes

TiBS now comes with a powerful tape repair program that makes it easy to recover from tape failures. For example, it can be used to recover a single failed tape volume with:

```
# taperepair –l pool –t tape_number –o offset
```

Upon successful completion, the failed tape volume will be recreated as a new tape volume. The failed merge process can then be re-tried. The failed tape volume will also be marked as invalid in the tape database and will not be used for future restore requests.

Recover from the backup client

Full Backups

Expire the current full pool and re-take the full backup. This can also be used to resolve `DRIVE_IDENT` errors.

```
# hostexpire –l pool –n client –p path -x
# tibs –l pool –n client –p path
```

Midlevel Backups

Midlevel backups may be optionally recovered by using the backup client and a network backup transaction. This may be the preferred method for some sites, because it does not require the loading of additional tapes. This feature can be used only if the `FAST_CACHE_RECOVERY_DIR` option in `tibs.conf` has been enabled. Before you can proceed you must also know:

1. The next lowest full or midlevel media pool.
2. The current incremental pool.

Recover from the last cache state of the next lower full or midlevel media pool:

```
# cacherecover –l lower_pool –n hostname –p path
```

Expire the current incremental (if any):

```
# hostexpire –l current_incremental_pool –n hostname –p path
```

Run a new backup for the current incremental pool:

```
# tibs –l current_incremental_pool –n hostname –p path
```

Re-run the merge for this pool:

```
# teramerge –l pool –n hostname –p path
```

Command Reference

Summary of Server Commands

Automation & Server Configuration	Backup Client Configuration	Media Management	Backup Operations
cachemgr	classaudit	cachemove	afsback
runtibs	hostadd	dlremove	afsbackup
stoptibs	hostaudit	tapedel	afsgen
tapemgr	hostdel	tapelabel	hostexpire
tibsdld	netaudit	tapemove	netbackup
tibstaped		tapescan	mergebackup
tibswd		tibsmnt	tibs
			teramerge
Restore Procedures	**Server Monitoring**	**Server Maintenance**	**Support Programs and Scripts**
afsrest	bigfiles	cacheclear	afsvcheck
afsrecover	cachestat	cacherecover	checkvol
filesearch	genreport	dlrecover	teranotify
fldblist	tapestat	taperepair	volscan
tibsrest	tibstat	tibssched	

Note: **All commands support standard flags documented here:**

- `-h` **Display online help**
- `-v` **Verbose mode**
- `-V` **Display program version**

Automation and Server Configuration

cachemgr: Automation program that is run at startup by the `tibswd` program. It monitors the backup cache and attempts to resolve potential cache overflows by balancing or clearing cache backup volumes. If the overflow cannot be resolved, an alert is sent to all defined administrators. This command should not be run from the command line.

runtibs: Backup server startup scripts usually run from `/etc/inittab` at server boot time and are also used to restart TiBS after some configuration changes.

-f	Fast start mode. If there are no cache backup volumes in the BUSY or WRITING state, this flag will ignore the cleanup of the backup caches.

stoptibs: Server shutdown script used to halt the backup server for some configuration changes.

tapemgr: Automation program is run at startup by the `tibswd` program. It looks for pending backup requests that have no associated tape mounted. A mount request is sent if no valid tape can be found. This command should not be run from the command line.

tibsrlm: TiBS Reprise License Manager™ (RLM) daemon. Started by `runtibs`, this process must be running when performing any type of backup. This command should not be run from the command line.

tibsdld: Automation program that is run by the `tapemgr` program. The program provides support for writing data to the optional disk library. This command should not be run from the command line.

tibstaped: Automation program that is run by the `tapemgr` program. The program provides support for writing data to tape and to the optional disk library. This command should not be run from the command line

tibswd: Backup server watchdog daemon that usually runs at startup from `runtibs`. Starts and monitors the `cachemgr` and `tapemgr` processes. This command should not be run from the command line. Instead, use `runtibs`.

Backup Client Configuration

`classaudit:` Auditing utility that generates an audit report on all hosts defined for backup by the network audit. There are currently no arguments for this command.

`-g`	Audit a specific backup group.

`hostadd` : Adds a new client backup volume to a backup class and backup group. The command will prompt the user for information required that was not entered on the command line.

`-F`	Force volume creation. Used to create a client volume even if the hostname is not currently valid.
`-o`	Override option. Used to define individual client volumes with separate rules, even if the client is defined to a default set that contains the volume name.
`-y`	Answer *`yes`* for automation. Prompt the user for verification before adding a new backup volume definition.
`-c class`	Backup class in which to add the volume **(required)**.
`-g group`	Backup group in which to add the volume **(required)**.
`-l location`	Specify the cache location for the backup volume. Only valid when creating the first backup volume for a client. Additional client definitions must reside in the same cache location, currently.
`-n client`	Name of the backup client to add **(required)**. The special client, *`default`*, defines a new default backup volume for the backup group.
`-p path`	Pathname of the client volume to backup **(required)**. The special path, *`default`*, defines all default backup volumes in the group for the client.
`-r rulefile`	Optional rule file. The default rule, *`none`*, is used if not specified.

`hostaudit`: Performs a partition level audit on a backup client by obtaining a list of currently mounted partitions and comparing them to the client's current class definitions. This is the primary program used to manage network clients for functions such as authentication, configuration, and debugging.

`-a`	Register a backup client. Only applies to backup clients that have not yet been authenticated.
`-c`	Show current client configuration file (`tibs.ini`).
`-d`	Debug host, prints the clients error log to `stdout`.
`-l`	List just the parts that the client has defined for backup, if any.
`-L`	List all parts defined for a client, including those defined for `omit` or `skip`.
`-o`	Omit check; make sure no parts are defined for this host. Doesn't contact client.
`-q`	Query host, prints client partition information in `df` format.
`-r`	Re-register a client. Deletes server authentication files before registering.
`-s`	Query host system type, prints clients current operating system and revision.
`-i ipaddr`	Used to communicate with TiBS clients that do not have a static IP address, when the client's current IP address is known.
`-n client`	Name of the client being audited **(required)**.
`-P port`	Use the specified port to communicate with the backup client. The default port is determined from the client's group definition. This option is typically only required by sites that are using alternate ports or have multiple boot backup clients.
`-u parameter`	Update a client configuration parameter (requires `-w value`).

Parameter	**Value**	
`MY_HOST_NAME`	hostname	Used to change the hostname in `tibs.ini` and re-authenticate a backup client.
`ENCRYPT_ENABLED`	0/1	(1 enables)
`IGNORE_SERVER_IP_CHECK`	0/1	(1 enables)
`MAX_LOG_SIZE`	10-2048	(Kilobytes)
`OTM_CACHE_SIZE`	10-20480	(Megabytes)
`OTM_WAIT_TIME`	1-30	(seconds)
`OTM_ENABLED`	0/1	(1 enables)
`OTM_THROTTLE_DISABLES`	0/1	(1 allows OTM backups to run faster)
`OTM_ALLOW_ACTIVE`	0/1	(1 allows)
`OTM_CACHE_PATH`	/c/program files/Teradactyl/tibs	UNIX style pathname of alternate OTM cache directory location.
`RUN_LOCAL_MODE`	0/1	(1 enabled local mode backups for clients located behind firewalls)
`VERBOSE`	0-10	(0 disables)

`-w value`	Value to which to set a client configuration parameter (requires `-u parameter`).

hostdel: Removes client backup volume(s) from class and backup group definitions. If all client volumes are being removed, the audit database is updated to omit the client from backup.

-F	Do not try to resolve hostname passed on the command line. Useful for DHCP clients and hostnames than may have been aliased to another hostname.
-q	Query shows what client volume(s) would be removed.
-y	Automation, answers *yes* to all confirmation requests.
-c class	Class from which to delete a volume definition. Required when deleting a default volume definition.
-g group	Group from which to delete a volume definition. Required when deleting a default volume definition.
-n client	Name of the backup client to remove.
-p path	Optional client path to remove. The command removes all volumes for the specified client if not specified.

netaudit : Runs a network audit on all sub-nets defined in `state/subnets.txt` and updates the network audit file `state/clients.txt`.

-a	Automation, unresolved audit information is left in `reports/networks/clients.nodef`.
-f	Force the removal of invalid hostnames. The default is to query or report. Useful when automating audits.

Media Management

cachemove: Program used to manually relocate all backup cache volumes for a client to an alternate location in order to balance the backup cache. Usually cache relocation is performed by the `cachemgr`.

-F	Do not resolve hostname. Useful for moving DHCP or clients.
-D	Do not move data. Only change the host's cache location. Used by `cacherecover` to relocate data to an alternate cache location. This is not to be used on the command line.
-y	Automation, answer *yes* to all confirmation requests.
-d path	Destination cache path **(required)** as defined in `state/caches.txt`.
-n client	Name of client to relocate **(required)**. All the client volumes are relocated.

dlremove: removes data not required for backup processing from the disk library to free needed space.

-F	Do not resolve hostname. Useful for a decommissioned host that has been aliased to another hostname.
-q	Query. Shows what volume(s) will be expired.
-y	Answer *yes* to confirmations for automation.
-D days	Expire volumes that are `days` old (calculated in seconds since Jan 1, 1970) from the current server time. Volumes that match other criteria will be expired only if they are at least this many days old.
-f seconds	Perform expires from this start time in `seconds` (since Jan 1, 1970). By default, `hostexpire` will generate the current time automatically. This parameter is, and should be used, for multiple expirations of multiple levels of full and midlevel backup volumes to keep volume expirations in synchronization.
-l pool	Tape `pool` from which to expire **(required)**.
-m megabytes	Limit the amount of data to megabytes of current data on tape. Used to control the flow of merge processing to smooth peaks in processing load on a daily basis
-n client	Name of the client host to expire.
-o offset	Expire volumes only from this tape offset. Typically used to remove a bad tape volume from the tape database (`-t tapeno` option required).
-p path	Pathname of the client volume to expire. If not specified, then expire all volumes found for the host (`-n client` option required).
-s seconds	Expire volumes that are `seconds` old (since Jan 1, 1970) from the current server time. Volumes that match other criteria will be expired only if they are at least this many `seconds` old.
-t tapeno	Expire volumes from this tape number. Typically used to remove a bad tape volume from the tape database (`-r` flag or `–o offset` option required).

tapedel: Tape and tape database management utility used to erase, recycle tapes, and delete tape definitions from the tape database.

-r	Recycle the tape. The contents of the tape database are removed. The next write to the tape will occur at offset 1.
-y	Automation, answer *yes* to all confirmation requests (use twice for complete automation)
-e device	Erase the tape mounted in device.
-l pool	Media pool to delete **(required)**.
-n tapeno	Tape number to delete **(required)**.

tapelabel: Tape label utility for adding new tapes to a media pool and reading tape labels.

-q	Query current tape label.
-R	Recursive call (used by automated `tapelabel` calls).
-y	Automation, answer *yes* to all confirmation requests.
-b blocksize	Override the default tape block size for this tape. The default block size is in `tibs.ini`. The range for `blocksize` is 16 thru 256. All volumes written to a single tape will use the same block size.
-c code	Barcode support. Place the bar code of the current tape onto the tape's label.
-l pool	Media pool in which to add a tape **(required)**.
-m device	The tape device containing a mirror tape to label.
-M code	Barcode support. Place the bar code of the mirror tape onto the mirror tape's label.
-n tapeno	Tape number in the pool **(required)**.
-t device	The tape device containing a tape to label **(required)**.

tapemove: Tape offsite management utility. As tapes are physically moved to an offsite location, this command can be used to update the offsite status in barcodes.txt.

-q	Query for tapes of a given status
-F	Mark tape as filled (requires –t tapeno)
-l pool	Media pool to update or query **(required)**
-t tapeno	Tape number
-m status	New tape mirror status **(either –m or -s is required)**.
-s status	New tape status **(either –m or -s is required)**.

tibsmnt: Tape and tape device management interface used by programs and operators to manage tape mount requests. Programs send e-mail alerts when requests cannot be completed. When run from the command line, operators will be prompted to load tapes that cannot be found. If no arguments are passed, the current mount state for all tape devices is displayed.

-a	Automation flag used by programs. Generates e-mail alerts for unsatisfied requests or errors.
-F	Forcibly unmount a tape from a drive for which the process ID is no longer valid. Used when a process dies and fails to release a tape device before exiting (e.g. after a power failure).
-H	Halt library automation for manual intervention.
-i	Initialize. Called from `tapemgr` at startup to reset all tape drives to an initial state (All tapes are taken offline from TiBS. If tape automation is used, the tapes are placed on an available shelf in the tape library; otherwise, tapes are left online to the operating system).
-I	Ignore tape label timestamps. Currently only issued by `tapescan`.
-q	Query the tape library for device ids, tape labels, and bar codes (ATLI only).
-c device	Manually clean a tape device
-d device	Disable a failed tape device
-e device	Enable a failed tape device
-f atli_id	Manually move tape from physical address. Requires `-t atli_id` option (ATLI only).
-g group	Device group in which to mount the tape. Typically used by programs that are trying to write data to tape. When the current tape fills, a new tape must be mounted to the same device group.
-j atlidev	Specifies the tape library device that will be used to perform the mount request (ATLI only).
-n tapeno	Tape number to mount. If not specified, then the lowest numbered available tape is used. Required for read or dismount requests.
-r pool	Mount tape for read. Requires `-n number` option.
-t atli_id	Manually move tape to physical address. Requires `-f atli_id` option (ATLI only).
-u pool	Dismount tape. Requires `-n number` option.
-w pool	Mount tape for write-only access. If no tape number is specified, the lowest numbered tape in the pool will be requested. If no currently labeled tape is available then a blank tape will be requested. If a specific tape is desired, use with the `-n tapeno` option.

tapescan: Used to verify the contents of a tape or recover the tape database for a tape. The media pool and tape number arguments are required. If no other arguments are specified, then the tape is scanned quickly and the contents of each tape volume header are compared against the backup server's current tape database.

-b	Rebuild the tape database and restore the file lookup database entries. This can take an extended period of time.
-i	Scan an individual tape volume.
-q	Queries the tape database and prints the list of current tape volumes.
-s	Performs a full verify of each volume on tape. This can take a long time.
-l pool	Media pool to which the tape being scanned belongs to **(required)**.
-o offset	Tape offset to start the scan (default=1).
-t tapeno	Tape number of the tape to scan **(required)**.

Backup Operations

afsback: Backup program used for AFS®. This command is called by `afsbackup` for cell backups.

-F	Force unchanged incremental backups to tape. The default is to backup volumes that have a `Last_Update` time stamp that has changed since the last backup.
-G	Generate AFS® `.backup` volumes during incremental backup. By default, `.backup` volumes are only generated during full backups.
-O	Override AFS volume id mismatches.
-q	Backup query. Shows what volumes are missing from the media pool.
-Q	Full query, check backup cache state
-r	Rollback the last backup time by one month. This option can be used to keep backups moving forward when AFS automation does not properly create `.backup` volumes.
-S	Perform network scan, don't actually generate a backup
-x	Perform a consistency check on each backup volume generated.
-c class	Name of the class to perform backups on (default = all classes).
-C cell	Perform backups on the specified cell (default = all cells).
-g group	Name of the backup group on which to perform backups (default = all groups, if a class is specified, then all `afs` groups in the class).
-l pool	Media pool in which to write backups **(required)**.
-n volume	Name of the read-write volume to backup. The `.backup` volume for the read-write volume is actually backed up.
-s host	Optional name of an AFS® fileserver to backup.

afsbackup: The script front end for all AFS® backup operations. This script has all of the useful flags that it passes on to *afsback* to perform the actual backup. To backup individual volumes use *afsback*.

-F	Force unchanged incremental backups to tape. The default is to backup volumes that have a Last Update time stamp that has changed since the last backup.
-G	Generate AFS® `.backup` volumes during incremental backup. By default, `.backup` volumes are only generated during full backups.
-w	Wait for all processing to end before exiting. Typically used in automation scripts.
-x	Perform a consistency check on each backup volume generated.
-c class	Name of the class to perform backups on (default = all classes).
-C cell	Perform backups on the specified cell (default = all cells).
-g group	Name of the backup group to perform backups on (default = all afs groups, if a class is specified, then all afs groups in the class).
-l pool	Media pool in which to write backups **(required)**.

afsgen: AFS® parts list generation program. Must be run BEFORE the generation of AFS® `.backup` volumes. Typically run as a `cron` job to update backup information about AFS® volumes and their last update times.

-g	Generate a report to send to defined backup administrators.
-C cell	Generate information for this AFS cell (default = all cells).

hostexpire: Expires the current backup volume on a client from a media pool. The expired backup volume is still valid for restore, but a new backup volume may be written to the media pool.

-A	Expire older tape volumes to archive. This results in volumes being moved from `entries.arch` from `entries.old` for the given media pool.
-b	Mark tape volume as bad. This is done by inserting the string "`badvol`." In front of the hostname. The `-t tapeno` and `-o` offset flags are required. This flag is currently only used when repairing a failed tape volume in the `buildtape.sh` example script. This flag should not otherwise be normally used.
-d	Don't recursively expire tag pools. The default is to expire child tag pools (if any) recursively.
-F	Do not resolve hostname. Useful for a decommissioned host that has been aliased to another hostname.
-L	Only delete disk library backup volumes (required `-r` flag). Used to remove disk library backup volumes w/o removing the corresponding tape volumes. Use of this flag requires the tape pool to be defined for both tape and disk backups.
-q	Query. Shows what volume(s) will be expired.
-r	Permanently remove tape volumes that are being expired. The contents of the tape database are removed for these volumes and they are no longer available for restore processing.
-R	Recursive call, doesn't check locks (should not be used from command line).
-x	Delete cache volumes for full media pools during automation. If this option is not set during automation, data in the cache will not be removed. If this option is not set from the command line, the user is prompted.
-y	Answer *yes* to confirmations for automation.
-D days	Expire volumes that are `days` old (calculated in seconds since Jan 1, 1970) from the current server time. Volumes that match other criteria will be expired only if they are at least this many days old.
-f seconds	Perform expires from this start time in `seconds` (since Jan 1, 1970). By default, `hostexpire` will generate the current time automatically. This parameter is, and should be used, for multiple expirations of multiple levels of full and midlevel backup volumes to keep volume expirations in synchronization.
-l pool	Tape `pool` from which to expire **(required)**.
-m megabytes	Limit the amount of data to megabytes of current data on tape. Used to control the flow of merge processing to smooth peaks in processing load on a daily basis
-n client	Name of the client host to expire.
-o offset	Expire volumes only from this tape offset. Typically used to remove a bad tape volume from the tape database (`-t tapeno` option required).
-p path	Pathname of the client volume to expire. If not specified, then expire all volumes found for the host (`-n client` option required).

-s seconds	Expire volumes that are `seconds` old (since Jan 1, 1970) from the current server time. Volumes that match other criteria will be expired only if they are at least this many `seconds` old.
-t tapeno	Expire volumes from this tape number. Typically used to remove a bad tape volume from the tape database (`-r` flag or `–o offset` option required).

`netbackup`: Performs network full and incremental backups by running a specified number of parallel TiBS backup processes.

-b	Process volumes by spawning a separate `netbackup` job for each host.
-O	Pass the drive identity override flag to all calls to TiBS. This should only be used in cases where the drive identity has changed on a UNIX partition due to a reboot or a change in the partition layout that has not changed the partition reporting the error.
-q	Query operation. Report on what backups still need to go to the media pool.
-Q	Query all, includes `roaming` and `afs` backup groups.
-c class	The class being backed up. The default action is to backup all classes that have the media pool defined.
-g group	Backup group being backed up. If not specified the default is all of the groups defined or if the class argument is used, then all groups in the specified class.
-l pool	Tape `pool` in which to backup **(required)**.
-L location	Only backup volumes from the specified cache location. The path is the same as the path defined in `caches.txt`
-n host	Run backups on a single client. If no path is specified, all volumes for the client are backed up.
-o host	Omit this host from backup processing. Used to separate out special hosts for alternative backup processing. This option may be used more than once to omit up to 32 separate hosts. The host name must match the hostname used to define the system on the backup server. This will usually be the fully qualified domain name.
-p path	Run backups on a specified path. If no client is specified, all clients that have the path defined are backed up.
-P count	Number of backup processes to run in parallel. The number of actual processes that run concurrently will be limited by the total number of backup processed licensed on the backup server.
-t seconds	Override the default `NETWORK_BACKUP_WAIT` time in `/etc/tibs.conf`. This is the amount of time that `netbackup` will wait between jobs for the same host. The default value of 60 seconds should be used when backing up Windows systems.
-w count	By default, `netbackup` will wait until all backup processing has completed. The option allows the program to exit when all jobs have been started and the number of processes still running reaches the count specified.

tibs: This is the primary network backup program used on the backup server.

-O	Override `DRIVE_IDENT` errors. Some operating systems do not maintain consistent drive identities when rebooted or when other partitions are added to the system. If it is known that a partition has not changed, this flag will allow for continued backup of the partition by overriding this error. **CAUTION**: If the partition has indeed changed, this will most likely result in a complete backup of the new partition. In this case, it is better to `hostexpire` the partition and perform a new full backup (see Recover from the Backup Client).
-q	Query operation. Report on what backups still need to go to the media pool.
-Q	Query all, includes `roaming` and `afs` backup groups.
-c class	The class being backed up. The default action is to backup all classes that have the media pool defined.
-g group	Backup group being backed up. If not specified the default is all of the groups defined or if the class argument is used, then all groups in the specified class.
-i ipaddr	Used to communicate with TiBS clients that do not have a static IP address, when the client's current IP address is known.
-l pool	Tape `pool` in which to backup **(required)**.
-L location	Only backup volumes from the specified cache location. The path is the same as the path defined in `caches.txt`
-n host	Run backups on a single client. If no path is specified, all volumes for the client are backed up.
-p path	Run backups on a specified path. If no client is specified, all clients that have the path defined are backed up.

mergebackup: Script used to automate running of parallel merge backups with `teramerge`.

-w	Wait for all merge processing and tape writing to complete.
-l pool	Tape pool in which to write backup volumes **(required)**.
-n count	Number of parallel `teramerge` processes to run. This number should not exceed the number of tape devices available for reading.

`teramerge`: This is the primary backup program used on the backup server. This program is used to run the full or midlevel merge process. This processing does not require interaction with the network or backup clients.

`-F`	Process only volumes that will be do not require data from previous tapes at this level.
`-q`	Query operation. Reports on what backups still need to be merged. Also reports state information for volumes that are not ready to be merged.
`-Q`	Query operation. Reports on what tapes will be required for merge. Useful for generating the complete list of tapes needed for a backup job.
`-x`	Verify new AFS® backup volumes after they are generated.
`-c class`	The class being backed up. The default action is to backup all classes that have the media pool defined.
`-g group`	Backup group that is being backed up. If not specified, the default is all of the groups defined. If the class argument is used then all groups in the specified class are defined.
`-l pool`	Tape pool in which to write backup volumes **(required)**.
`-n host`	Run `teramerge` on a single client. If no path is specified, then all volumes for the client are backed up.
`-p path`	Run `teramerge` on a specified path. If no client is specified, then all clients that have the path defined are merged.
`-t tapeno`	Only merge volumes from this `tapeno`.

Restore Procedures

afsrest: Program for restoring individual AFS® volumes. By default the data is restored to the same location as the existing volume with a `.restore` extension.

-f	Fast run, don't check tape library or disk library
-F	Force the overwrite of any existing volume(s). By default, `afsrest` will fail if the destination volume already exists in the cell.
-i	Restore data only from the most recent tape or cache backup volume found.
-o	Restore to the original volume name. By default, a `.restore` extension is added to the volume name, unless an alternate volume name is specified. Add the `-f` flag to overwrite any existing volume.
-q	Query restore. Used to find out what tapes perform a restore request.
-Q	Query all. List all tape volumes, most recent first, that are available for restore.
-a volname	Alternate restore volume name.
-C cell	Restores the AFS volume to this cell. This is required if more than on cell is supported.
-e extension	Add a `.extension` to every restored AFS® volume. Do not specify the dot (.) in the extension used.
-n volname	The name of the read-write volume to restore from backup.
-r server	Redirect the restored volumes to an alternate fileserver.
-p partition	Redirect the restored volumes to an alternate vice-partition. Cannot be used when restoring an entire fileserver.
-t time	Restore time in `YYYY/MM/DD/hh/mm/ss` format (`YYYY` required).

afsrecover: Disaster recovery program for fileservers. Used to restore vice-partitions, fileservers, or an entire cell. This assumes that the Volume Location Database (vldb) is current. If you are recovering an entire cell, make sure the vldb is restored first.

-q	Query the restore database for a list of tapes to complete the request.
-Q	Show the order that data will be restored.
-a partition	Redirect data to an alternate vice-partition (requires `-p partition`).
-C cell	Recover the AFS cell. Required if more that one cell is supported on the backup server.
-e extension	Add a `.extension` to every restored volume. Do not specify the dot (.) in the extension used.
-l pool	Use the pool for all full restore requests.
-p partition	Restore all read-write volumes for the partition. If no `-s server` argument is supplied, then all servers containing the partition will be selected.
-r server	Redirect data to an alternate fileserver (requires `-s server`).
-s server	Restore all vice-partitions on a fileserver. If the `-p partition` argument is supplied, then only the read-write volumes on that partition will be selected.

`filesearch`: Script that scans the backup server's tape databases looking for the location of files or directories on tape. The search can be limited by the client's hostname, the pathname, and query string. The more limited the search parameters, the faster the search time.

`-c`	Count the number of tape database searches that would be performed.
`-d`	Limit the search to directory names only.
`-e`	Exact match. Process the search as case sensitive.
`-f`	Limit search results to files only (not directories).
`-n client`	Hostname of the backup client to search.
`-p path`	The pathname of the backup volume to search.
`-s string`	Partial or complete file name to search (no wildcards).

`fldblist`: Program that prints file lookup database information in human readable form.

`-a`	Print only active file information
`-b`	Print only bad file information.
`-c`	Print file/directory count information.
`-d`	Print only directory information.
`-F`	Print only valid file information.
`-o`	Print only open file information.
`-r`	Print only file reference information for files not on this tape.
`-s`	Print summary header information. Used by programs and scripts that gather aggregate information such as `tapestat` and `cachestat`.
`-x`	Ignore file reference (REF) information in incremental (non-full) tape volumes.
`-f file`	Name of the `.fldb` or tape database file to scan.

`tibsrest`: The primary restore program used on the backup server.

`-c`	Only restore data located in the backup cache volume.
`-d`	Delete client files that are not part of the restore. Removes any files that are not part of the restored image. Use with caution. Not needed if the restore directory is already empty.
`-f`	Fast query. Used with Automated Tape Library Interface to bypass the check for tape availability within defined tape libraries.
`-F`	Force restore of an invalid hostname. Do not resolve hostname of backup client.
`-i`	Run an incremental restore only. Do not restore the contents of the full backup. Used for single file or directory restores.
`-q`	Query restore. Used to find out what tapes perform a restore request.
`-Q`	Query all. Lists all tape volumes, most recent first, that are available for restore.
`-R`	Recovery option. Only restore files that are more recent than the files found on the client. Typically used for server disaster recovery, when the restore process is done in reverse to prevent the overwrite of already restored files.
`-x`	Used to recover from a broken tape chain. This flag will enable restore to make the best guess on which tape volumes to use. Upon completion, restore will display a list of files it was unable to restore.
`-X`	Restore from tape only. Do not use the backup cache for restore.
`-y`	Used for automation. Specifically, to allow automated recovery to the original backup location. By default, `tibsrest` will prompt before overwriting data from the original location.
`-a path`	Used to restore data to an alternate location.
`-I ipaddr`	Used to communicate with TiBS clients that do not have a static IP address, when the restore client's current IP address is known.
`-l logfile`	Output log file used for debugging. The default is `stdout`.
`-n client`	Name of the backup client to restore taken from **(required)**.
`-p path`	Access path used to create the backup volume **(required)**.
`-P port`	Contact client at an alternate TCP port number. This overrides a port found by looking up the service using `getservbyname()`, or the default port, 1967.
`-r client`	Restore to an alternate client.
`-s path`	Relative path of sub directory to be restored (no leading '`/`').
`-t time`	Point in time to restore data. The format is `YYYY/MM/DD/hh/mm/ss`. At least `YYYY` must be specified.

Server Monitoring

`bigfiles`: This script generates a listing of the larges files for each current backup for the specified media pool. It then sorts and reports on the largest files it finds.

`-a`	Include information about AFS volumes (may be slow for large AFS cells).
`-l pool`	Tape `pool` to report on **(required)**.
`-n count`	Report on the top count files (default=20).

`cachestat`: Return utilization information about the backup cache. This information can be used to track and optimize backup performance.

`-s`	Print summary cache statistics in `df` style format.

`genreport`: Automation script to send summary information on daily backup progress to defined operators and administrators. It may also be run manually. Run with no arguments to generate a full report.

`-a`	Generate a full report (include all additional reports).
`-A`	E-mail reports to all administrators, including `REPORT_ADMINS`.
`-b`	Include top 20 files report from incremental backups.
`-c`	Include cache statistics report.
`-i`	Ignore network timeout errors.
`-n`	Include network audit report.
`-p`	Include partition (class) audit.
`-q`	Test reporting. Logs are not updated and the report is displayed to the screen.
`-r`	Generate client and server report summary.
`-s`	Include skipped parts report.
`-t`	Include tape usage report.
`-x`	Do not e-mail report.
`-m string`	Assign a name for the report (default=pool-Backup-Report).
`-C report`	Run custom reporting before processing other reports. This allows a site to monitor specific error conditions and report additional local site information (such as room number or system owner). Typically, once a custom report is written for a specific error condition, it can then be filtered out of the standard report.

`tapestat`: Show tape utilization information for the specified media pool.

`-a`	Generate only AFS® statistics.
`-A`	Include data from archived tapes (`entries.arch`). By default, `tapestat` only reports on tapes that are part of the current backup set (`entries.old/entries.txt`).
`-b`	Brief. Do not include zero results
`-c`	Only include current tape volumes. Does not include statistics about expired tape volumes.
`-n`	Include network and cache statistics for each tape found.
`-s`	Print by client information in gigabytes.
`-w`	List tapes available for write access, a + preceding a tape number indicates a new tape.
`-x`	Do not include AFS® statistics.
`-l pool`	Tape `pool` to report **(Required)**.
`-t size`	Report only those clients that consume `size` or greater gigabytes of tape.
`-u size`	Report only those clients that consume `size` or less gigabytes of tape.

`tibstat`: Show current backup job status and backup volume state information. Shows all status info if no arguments are passed.

`-b`	Show busy volumes.
`-j`	Show process status.
`-p`	Show pending volumes.

Server Maintenance

cacheclear: Clears busy volumes from the cache caused by improper processing.

-a	Clear write flags (used by startup mode only).
-F	Do not try to resolve hostname passed on the command line. Useful for DHCP clients and hostnames than may have been aliased to another hostname.
-q	Query, shows what volumes would be cleared.
-n host	Only clear volumes on the specified host (default=all hosts).
-p path	Only clear volumes that match the specified path. (default = all paths for host(s)).

cacherecover: Restores data from tape directly to the backup cache.

-F	Do not try to resolve hostname passed on the command line. Useful for DHCP clients and hostnames than may have been aliased to another hostname.
-k	Keep midlevel data in the cache. By default, `cacherecover` will only restore database information when recovering data from midlevel media pools. This flag overrides this behavior (see Recovering from a Failed Tape Merge).
-K	Keep midlevel file lookup database information in the cache. This can be used to recover file lookup database entries not recovered by `tapescan -b` when recovering the backup server from tape (see Backup Server Recovery).
-N	Override network only recovery for fast cache recovery. By default, TiBS will mark a recovered cache database to require a new network backup. This option disables this feature.
-q	Query to see what volumes will be recovered.
-r	Use the cache recover database to recover the cache.
-R	Restart the recovery. Skip over any backup volumes that have already been recovered. Typically only used when a previous recovery command has failed and appropriate action has been taken to work around the failure (see Cache Recovery Failures).
-s	Scan the cache but does not perform a cache recovery.
-x	Disable use of the fast cache recovery information. By default, `cacherecover` will attempt to use information from the fast cache recovery directory when available. This option is used when recovering failed tape volumes (see Recovering from a Failed Tape Merge).
-X	Only recover if an exact time stamp match is found (requires `-T time` option).
-a location	Redirect restore to an alternate cache path. Cannot be used if the entire cache is being restored.
-c location	Only recover volumes only for the cache location.
-l pool	Recover volumes from the specified media pool. The most recent volume at this or a higher level is used to recover the cache. This option is required when using the `-r` flag.
-n host	Only recover volumes for the specified host.
-p path	Only recover volumes for the specified path (requires `-n host`).
-t number	Only recover volumes for the specified tape number (requires `-l pool`).
-T time	Time in `YYYY/MM/DD/HH/MM/SS` format. By default, `cacherecover` will use the most recent cache recovery or tape volume available. This option allows data recovery to the cache from older cache recovery or tape volumes.

`dlrecover`: Restores data from tape back to the disk library.

`-F`	Do not try to resolve hostname passed on the command line. Useful for DHCP clients and hostnames than may have been aliased to another hostname.
`-i`	Used to inform `dlrecover` to only restore the most recent incremental backup for each `host:partition` for the tape pool being recovered. This is useful when recovering an entire disk library and the most recent backup for each level must be recovered to continue merge processing from disk.
`-q`	Query to see what volumes will be recovered.
`-R`	Restart the recovery. Skip over any backup volumes that have already been recovered. Typically only used when a previous recovery command has failed and appropriate action has been taken to work around the failure (see Cache Recovery Failures).
`-s`	Scan the disk library but does not perform a disk library recovery.
`-X`	Only recover if an exact time stamp match is found (requires `-T time` option).
`-a location`	Redirect restore to an alternate cache path. Cannot be used if the entire cache is being restored.
`-c location`	Only recover volumes only for the cache location.
`-f time`	Time in `YYYY/MM/DD/HH/MM/SS` format. By default, `dlrecover` will attempt to recover all volumes for a tape pool. This option allows data recovery to the disk library for only those volumes that are more recent than the specified time.
`-l pool`	Recover volumes from the specified tape pool **(required)**.
`-n host`	Only recover volumes for the specified host.
`-p path`	Only recover volumes for the specified path (requires `-n host`).
`-t number`	Only recover volumes for the specified tape number.
`-T time`	Time in `YYYY/MM/DD/HH/MM/SS` format. By default, `dlrecover` will attempt to recover all volumes for a tape pool. This option allows data recovery to the disk library for only those volumes that are older than the specified time.

`taperepair`: Repair some or all of a failed tape.

`-q`	Query, do not perform repair operations
`-B`	Mark older volumes as bad
`-W`	Update repair volume write times
`-R`	Create the repair cache directory
`-M`	Perform repair merges from the repair cache
`-l pool`	Tape pool of tape(s) to repair (required)
`-o offset`	Repair a single offset (requires -t tapeno)
`-r tapeno`	Recover phase from a single tape (requires -R or -M)
`-s offset`	Repair tape starting at offset (requires -t tapeno)
`-e offset`	Repair tape ending at offset (requires -t tapeno, -s offset)
`-t tapeno`	Repair part or all of a single tape

tibssched: Used to rebalance the backup load after significant backup server down time (days).

-q	Query mode, run scheduler prechecks only
-t	Repair tape ending at offset (requires -t tapeno, -s offset)
-l pool	full media pool (required)

Support Programs and Scripts

afsvcheck: This program is called internally by the TiBS program when the `-c` flag is used to ensure that a file is a correct `vos` dump image.

-s	Short check, just read the dump volume header information.
-t	Check for empty file tags in full backup volumes.
-f filename	Pathname of the file that contains an AFS® *vos dump* **(required)**.

checkvol: Utility program that checks the consistency of the merge state of backup volumes. This program also implements a new prototype by-size backup scheduler. Please consult with Teradactyl before using the prototype by-size scheduler.

-a	Perform checks on defined AFS volumes.
-d	Disable incremental media pool checks
-F	Do not resolve client hostname if specified from the command line.
-s	Enable prototype by-size backup scheduler. The prototype scheduler compares this size of backups at each backup level and determines if the next lower backup should be scheduled. The determining factor is the size of the current backup, as a percentage of the size of the next lower level backup. The size of the current backup must be at least 125% of the next lower level backup.
-q	Query mode. Used in conjunction with the by-size scheduler to test what volumes would be scheduled.
-c class	Backup class to examine.
-g group	Backup group to examine.
-i pool	Incremental media pool. This option is required for the prototype by-size scheduler. More than one incremental media pool may be specified.
-l pool	Full media pool to examine **(required)**.
-n hostname	Optional backup client hostname to examine.
-p path	Optional backup pathname to examine.
-P percent	The percentage to apply when comparing the size of backup volumes using the prototype by-size scheduler. This value must be at least 125. If the backup volume being examined is at least this `percent` of the next lower level backup, then the by-size scheduler will schedule the next lower level backup.

teranotify: Script used to alert backup administrators of problems and tape mount requests.

-p	Print to screen and do not send e-mail alerts.
-f filename	Include filename in e-mail message.
-m string	Short message to send.
-s string	E-mail subject.

volscan: Scans backup databases and prints status information about all files and directories that it finds.

-F	Full volume verify all files are present
-r	Read header information only.
-R	Research flag, displays the largest inode found.
-s	Scan the entire cache volume, including the file stream. Used to verify cache volume integrity.
-S	Scan the entire cache volume, including the file stream. Includes checksum analysis for a detailed consistency check.
-c cachepath	The absolute directory path for the location of the dbfile to scan. Required when performing a scan (`-s` or `-S` options)
-f dbfile	Default database file to scan (default=`stdin`).
-l logfile	Output log file (default=`stdout`).

Client Commands

tera: Client initiated backup utility.

-a	Authenticate client host with the backup server defined in the client's `tibs.ini` (`PRIMARY_AUTH_SERVER=server.domain`). Clients must be authenticated before backup/restore processing can begin. You can also authenticate a client from the backup server using the `hostaudit` command. You must use the fully qualified hostname of the backup client in the `MY_HOSTNAME` field in the client's `tibs.ini` file.
-A	Same as `-a` flag except that the `MY_HOSTNAME` field does not have to be a fully qualified hostname. Useful when authenticating clients that do not have a static IP address. This type of authentication can only be performed from the backup client.
-l	Run local mode backups. Used to perform backups when the backup server cannot communicate with a client's backup service. For example, if the client is behind a firewall. The local mode backup will allow the client to run the backup to the server. The user must be root on UNIX systems or have Administrator privileges on Windows systems. Local mode backups may also be enabled using the `USE_LOCAL_MODE=1` setting in the client's `tibs.ini` file.
-q	Queries the backup server to see what volumes need to be backed up.
-Q	Queries the backup server for backup status, including part definitions and the last backup time.
-w	Script Mode, wait for user input before exit.
-D path	Location of client configuration files. Required when configuring a UNIX client to use tcp wrappers.
-p path	Path of the client volume to be backed up.
-P port	Contact server at an alternate TCP port number. This overrides a port found by looking up the service using *`getservbyname()`*, or the default port, 1968.

Technical Notes

These technical notes describe how TiBS performs under certain conditions. It is important that administrators understand how data is backed up and will be restored. If you have any questions or concerns about any of these notes, feel free to contact Teradactyl® for assistance.

Backup and Restore of Hard Links

If an entire disk partition is not backed up as a volume, it is possible to backup a file with multiple hard links, some of which are not in the backup volume. It is best to backup full partitions to avoid this. The restore of a sub-directory can generate the same error if all of the hard-links are not in the sub-directory to be restored. In this instance, a warning message is printed to indicate that the hard link status has changed for a file. This has no impact on disaster recovery for complete file systems.

Server Reserved Characters

The following characters are currently reserved as separators in the server databases and cannot be used as characters in pathnames, hostnames, etc.

|	The "or" or "pipe" character.
\n	The new line character.
\r	The carriage return character (Windows®).

Server Reserved Words

afs	Reserved hostname for AFS® backups.
file	Reserved rule file name.
none	Reserved rule file name.
omit	Reserved rule file name.
persist	Reserved rule file name.
skip	Reserved rule file name.
unused	May not be used as a valid media pool.

Restore of UNIX® Special Files

The restore of special devices is not supported to alternate operating system types. Most devices are located on the root file system. It does not make sense to restore a root file system from one operating system type to another. The restore process will attempt to restore special files, but the results are unpredictable. This does not include symbolic or hard links. They will be restored to an alternate system if they are supported, including NTFS under Windows®.

Macintosh OSX Recovery Notes

TiBS supports the backup of UFS, HFS+ and HFS file systems. Because of the differences in the structure and meta data that is backed up on these files systems, some types of recovery can fail. Restore procedures are currently supported back to the original file system type. Other types of restores can be attempted, but they may fail with the following problems:

Restore from/to	Problems/Un-restored data
HFS to HFS+	Supported.
HFS to UFS	Unsupported. The UFS restore client does not currently recognize traditional resource fork (HFS style) information stored under HFS.
HFS+ to HFS	Unsupported. The folder name limit of 31 bytes for HFS will cause a restore to fail if the limit is exceeded. The file name limit of 31 for HFS bytes will cause files to be skipped during the restore process if the limit is exceeded. Restore will indicate which files were not restored.
HFS+ to UFS	Unsupported. The UFS restore client does not currently recognize traditional resource fork (HFS style) information stored under HFS+.
UFS to HFS	Unsupported. HFS will not recognize resource information stored under UFS. The folder name limit of 31 bytes for HFS will cause a restore to fail if the limit is exceeded. The file name limit of 31 for HFS bytes will cause files to be skipped during the restore process if the limit is exceeded. Restore will indicate which files were not restored.
UFS to HFS+	Supported. HFS+ will recognize resource information stored under UFS.

Restore of HFS/HFS+ backups to TiBS Windows® and UNIX® clients is not currently supported. Support for restoring the file data for HFS/HFS+ backups to alternate operating and file systems will be introduced in the next client release.

Glossary of Terms

AFS: The Andrew File System is a network file system that is made up of many client data volumes that may be viewed over the Internet from a properly authenticated client.

Authentication: A process that is typically performed at install time which registers a backup client with a backup server. Once the registration completes the client may only communicate with the server to which it was authenticated.

Backup Cache: A group of hard-drive partitions located on a backup server that are used to store backup volumes.

Backup Class: A grouping of backup volumes which are to be managed by a common set of backup media. Classes are useful for sites that have more than one set of backup requirements.

Backup Volume: A single file located in the backup cache and/or one or more backup tapes. The data in a backup volume is obtained from the full or incremental backup of a client volume.

Client: A network computer with both the data that needs to be backed up and the TiBS Scaleable Network Backup Client installed.

Client Volume: The directory of client data which is backed up. All information in the directory will be backed up unless an optional rule file is used to eliminate unwanted data.

Default Backup Volume: A special backup volume definition which uses the reserved hostname, default, to specify a commonly backed up partition or pathname within a backup group. Clients defined in the backup group may use the special pathname, default, to backup all of the default volumes defined for the group.

Device Group: This is used to specify more than one tape drive on a single device chain (e.g. SCSI chain). Tape drives in the same device group cannot be written to at the same time.

Disaster Recovery: A process of restoring some or all of the data for one or more computer systems from backup due to catastrophic failure. Typically requires significant planning before the event so that all information required for the recovery process can be made available at recovery time.

Expire: A process that removes the client backup volumes from the current list for a given media pool. The backup volumes remain valid for restore requests. Once a backup volume is expired, a new copy of the backup volume may be written to the media pool.

Partial Cumulative Backup: A specialized form of Incremental Backup which backs up the files that have been modified or created since the most recent backup at the same level or lower (level n or lower).

Cumulative Backup Volume: backup the files that have been modified or created since the most recent lower level backup (level n-1 or lower)

Full Merge Backup: A process that combines backup volume information from the backup cache and tapes to produce a new full backup volume without interacting with the backup client.

Full Network Backup: A process that copies all data and meta data from a client to a backup volume.

Full Backup Volume: A file or group of files, which contains a complete copy of a client volume.

Group: A subset of a class, which are primarily used to identify groups of similar backup clients (e.g. Solaris clients).

Hard Link: The number of file system directory references to a single file.

Incremental Backup: A process that copies data and meta data that has changed since the last backup of a client volume to a backup volume.

Incremental Backup Volume: A backup volume which contains cumulative data changes since the last full backup.

Meta Data: Additional information associated with a file or directory such as ownership, security, or last modify time.

Network Audit: A procedure that scans a defined network for all currently defined Internet Protocol (IP) addresses. The addresses are compared with a network audit database and discrepancies such as new or removed host names are reported.

OTM: Columbia Data Products' Open Transaction Manager™ for Windows®. This is used to enable TiBS to backup open files and Windows NT/2000/XP/2003 operating systems.

Registry: A Windows® information database used to store information for the Windows® environment. This information can and should be backed up with TiBS.

Rule File: A file containing the relative pathnames of file and/or directories that are not to be backed up. A rule file may be assigned to any backup volume definition once it is created.

Server: Network computer with tape drives, a backup cache, and the TiBS, TiBS Lite, or TiBS Small Business Edition True incremental Backup System® Server installed.

Tape Drive: A removable media device which is used to store backup volumes for disaster recovery and/or offsite storage.

Tape Label: The name which is given to a specific tape, or tape set, in a media pool. The tape label includes the media pool and the tape number (e.g. `pool.1`).

Tape Pool: A collection of tapes used to store backup volumes. The tapes are labeled from 1 to the maximum number of tapes required to store all of the backup volumes.

TeraMerge®: A Teradactyl® unique technology which allows two backup volumes to be merged together to form a single backup volume. The single backup volume represents a full or incremental backup volume that is identical to one that could have been produced directly from a client, but was not, to reduce load on both the network and the client.

TiBS: An acronym for the True incremental Backup System®.

Traditional Level Backup: Common backup method that employs up to nine (9) backup or dump levels. Each level backs up data that has changed since the time of the most recent lower level backup.

Notes

Notes

Notes

Notes

Notes